Einführung in die Expertensystem-Technologie

Karsten Hartmann

19.12.2010

Vorwort

Das vorliegende Lehrbuch fasst die Erfahrungen zusammen, die ich im Laufe meiner industriellen Tätigkeit und im Zusammenhang mit meinen Vorlesungen zur Künstlichen Intelligenz am Fachbereich Informatik der Hochschule Merseburg gemacht habe. Es wendet sich an Studierende der Künstlichen Intelligenz und beansprucht nicht das Verdienst, die gegenwärtigen wissenschaftlichen Erkenntnisse im Bereich der Expertensysteme oder der wissensbasierten Systeme darzustellen. Vielmehr soll es den Studierenden neben der Darstellung der verbreitet eingesetzten Expertensystemtechnologie auch die Erfahrungen eines Entwicklers in diesem Gebiet darlegen.

Aus der Sicht eines Informatikers, der sich vor 16 Jahren aus der Entwicklung zurückgezogen hat und sich seitdem der Lehre zuwendete, empfand ich es erstaunlich, dass die Forscher in diesem Gebiet nur wenig wirklich Neues gefunden oder dargestellt haben. Obwohl ich fast ein halbes Jahr intensiv recherchierte und die angegebene Literaturstellen nur eine mir sinnvoll erscheinende Auswahl darstellen, ist der Erkenntnisstand der Abteilung in der ich damals arbeitete heute schon fast als modern bezeichnet zu werden.

Mag nun sein, dass viel der Erkenntnisse wie damals auch als industrielle Geheimnisse angesehen werden, mag aber auch sein, dass das Interesse an Expertensysteme heute eingebrochen ist, weil mittlerweile die Kosten deutlich geworden sind.

Für die Unterstützung bei der kritischen Überarbeitung des Skripts möchte ich mich bei den Studierenden des Master-Studienganges Informatik und Kommunikationssysteme der Eintrittsjahre 08, 09 und 10 bedanken. Für das Korrekturlesen bedanke ich mich bei Frau Anke Tallig. Für weitere Hinweise bin ich dankbar.

Inhaltsverzeichnis

1 Einleitung 1
2 Grundlagen 5
2.1 Geschichtliche Einordnung 5
2.1.1 Der Begriff „Künstliche Intelligenz“ 5
2.1.2 Turing vs Searle 6
2.1.3 Summerschool for Artificial Intelligence 9
2.1.4 Unterteilung der KI 10
2.1.5 Teilgebiete der Künstlichen Intelligenz 11
2.2 Wissensbasierte Systeme 13
2.3 Vom WBS zum XPS 16
2.3.1 Definition eines Expertensystems 18
2.4 Geschichte der XPS 21
2.4.1 XPS im Verlauf der Jahrzehnte 22
2.5 Einsatzbereiche von XPS 26
2.5.1 klassische Betrachtungsweise 26
2.5.2 Einteilung nach Zielsetzung und Motivation 31
2.5.3 Ableitung einer Aufteilung von XPS aus dem Bereich Diagnose 32
2.5.4 Expertensysteme als eine Ausprägung von Informationssystemen 34
2.5.5 Netzwerkbasierte XPS 35
3 Von der Daten- zur Wissensverarbeitung 37
3.1 Daten/Information/Wissen 37
3.1.1 Begriffsbestimmung 37
3.1.2 Austausch von Wissen 42
3.1.3 Der Wert von Information 45
3.2 Wissen 46

3.2.1 Philosophischer Ansatz von Ryle u.a. 46
3.2.2 Unterteilung nach Gottlob 49
3.2.3 Erweiterung des Wissens-Begriffs nach Lehner 50
3.2.4 Die Rolle der Erfahrung bei der Anwendung von Wissen 53
3.2.5 Wissensrepräsentation 54
3.3 Wissens-Management 56
3.3.1 Definition und Anwendungsbereiche 56
3.3.2 Rollen im Wissensmanagement 59
3.3.3 technologischer und humanorientierter Ansatz von WM 60
4 Aufbau eines Expertensystems 61
4.1 Überblick und Einordnung 61
4.1.1 Einteilung der Systeme aufgrund der Interaktion Mensch-Umwelt 61
4.1.2 Problemfelder 63
4.1.3 Überblick über die Komponenten eines XPS 65
4.1.2 Shells/Tools 68
4.2 Die Wissensbasis 70
4.2.1 Grundsätzliche Bemerkungen zu Regeln und Frames 71
4.2.2 Frames 73
4.2.3 Hybride Repräsentationsformen 76
4.2.4 externe und interne Repräsentation 77
4.3 Die Dialogkomponente 79
4.3.1 Einstieg und prinzipieller Aufbau einer Oberfläche 81
4.3.2 Problemlösungsverlauf 82
4.3.3 Erklärung 88
4.3.4 XPS als Handbuch 93
4.3.5 Notizbuchfunktion 95
4.3.6 Wissenspflegemodus 98

4.4 Die Erklärungskomponente ... 103
4.4.1 ein kurzer Blick in die Geschichte ... 104
4.5 Die Akquisitionskomponente ... 106
4.6 Die Problemlösungskomponente ... 110
4.6.1 Generelle Regelverarbeitung ... 111
4.6.2 Spezielle Regelverarbeitung ... 118
4.6.3 Frameverarbeitung ... 128
4.6.4 Verarbeitung innerhalb hybrider Systeme ... 132
4.6.5 Verteilung der Problemlösung ... 133
4.6.3 Verarbeitung von unsicherem Wissen ... 137
4.6.4 Rücknahme von Eingaben und Belegungen ... 140
4.7 Problemumfeld ... 143
4.7.1 Menschliche Problemlösung ... 143
4.7.2 Fallbasiertes Problemlösen ... 147
4.7.3 Unterschiede zwischen XPS und DB ... 150
4.7.4 Beispiel eines verteilten XPS ... 152
5 Das rote Wunder ... 155
5.1 Vorgehen innerhalb der Diagnose ... 155
5.2 Aufbau der Station ... 157
5.3 Wissensrepräsentations-Sprache ... 159
5.4 Problemlösung ... 160
5.4.1 Vor-, Verlaufs- und Nachbedingungen ... 164
5.5 Beispielwissensbasen ... 166
6 Knowledge Engineering ... 173
6.1 Wissenserhebung ... 178
6.1.1 Der Wissensingenieur ... 180
6.1.2 Der Experte ... 181

6.1.3 Erhebungsmethoden ... 183
6.2 Wissensmodellierung ... 190
6.2.1 mentale Modellbildung ... 191
6.2.2 Ontologie ... 194
6.2.3 Wissensrepräsentation ... 196
6.2.4 Strategien für die Problemlösung ... 198
6.3 Wissensintegration ... 201
6.3.1 Verwendung eines Parsers ... 202
6.3.2 Verwendung einer Akquisitionskomponente ... 204
6.3.3 Modellierungswerkzeuge ... 206
6.4 Wissensevaluierung ... 209
7 Schlussbemerkung und Ausblick ... 213
8 Literatur ... 217
Anhang A: Bauteilebaum der Transferstraße „Rotes Wunder“ ... 235
Anhang B: Syntax der verwendeten Wissensrepräsentationssprache ... 240
Bilderverzeichnis ... 243

1 Einleitung

In den 80ziger Jahren des vorigen Jahrhunderts kam das Modewort „Expertensysteme“ auf. Wie ein Blick in die damaligen Artikel der Computerwoche zeigen wurden mit diesen Systemen große Hoffnungen verbunden /Computerwoche 84/. Man war der Meinung, dass XPS möglicherweise in der Lage wären „komplexe Probleme im Produktionsbereich zu lösen“, man erwartete „eine größere Veränderung des Geschäfts oder … erhebliche Produktivitätsverbesserungen der Mitarbeiter“. Außerdem würden viele Endbenutzer die Fähigkeiten des Expertensystems als eine Erweiterung ihrer eigenen Fähigkeiten bei der Ausübung ihrer Arbeit betrachten.

Diese Aussagen wurden nach dem oben genannten Artikeln von einem Mitarbeiter der Firma Digital Equipment Corp. nach der Entwicklung eines einzigen ernsthaften XPS getan. Das System XCON (Expert VAX Systems Configurator) diente dabei zur Konfiguration von VAX 11/80-Computersystemen und war rein regelbasiert. Im Gegensatz zu einem eingearbeiteten Mitarbeiter, der eine Konfiguration einer VAX-Anlage in 20 bis 30 Minuten erledigte, konnte XCON die Aufgabe in weniger als einer Minute erledigen. Dabei berücksichtigte das System nach /Computerwoche 84/ mehr als 2000 Regeln.

Die Entwicklung XCONs begann als Gemeinschaftsprojekt von DEC und der Carnegie Mellon University im Jahre 1979. Schon bei der Entwicklung dieses Systems zeigte es sich, dass der Benutzer solcher Systeme stark in den Entwicklungsprozess einbezogen werden muss. „Dieser werde von Anfang an, also von der Problemskizzierung über die Erstellung einer Prototyplösung, über das Stadium, in dem Fachwissen gesammelt wird, bis hin zum fertigen Produkt am Entstehungsprozeß beteiligt.“ /Computerwoche 84/

Schon damals war aber klar, dass sich die XPS von anderen Systemen wie Datenbanken und Management Informations Systemen (MIS) klar unterscheiden. „Allerdings dürfen diese Systeme nicht mit Datenbanken oder Management-Informations-Systemen verwechselt werden.„/Mörler 84/ Das Besondere eines XPS liegt darin, dass das Problemlösungswissen der Experten in Form einer Wissensrepräsentation, etwa Regeln oder Frames, abgebildet und in einer Wissensbasis abgespeichert wird. Ein besonderer Problemlöser greift zur Lösung eines bestimmten Problems auf diese Wissensbasis zu, stellt diese dem Benutzer dar und erklärt diese Lösung auch.

„Ähnlich wie in Datenbanken durch die Trennung von Daten und Datenverarbeitung hohe Produktivitätsfortschritte erzielt wurden, geschieht dasselbe auf höherer Ebene in Expertensystemen durch die Trennung von Wissen und Wissensverarbeitung.“ /Mörler 84/

Der kommerzielle Nutzen von XPS wurde Mitte der 80ziger Jahre durchaus euphorisch gesehen. DEC ging von einer Einsparung von zweistelligen Millionenbeträgen durch die Einführung von XCON aus. „Die Marktbeobachter erwarten geradezu eine Explosion der Umsätze von rund 80 Millionen Dollar für 1984 auf 2,5 Milliarden Dollar in 1993. „ Gefragt wurde in diesem Artikeln auch ob sich die deutsche Wirtschaft an der Verteilung dieses Marktkuchens beteiligen würde.

Im Jahre 1985 schloss ich mein Studium an der technischen Universität Berlin ab und nahm eine Stelle bei der Innovationsgesellschaft für fortgeschrittene Produktionssysteme in der Fahrzeugindustrie (INPRO) an. Meine Aufgabe war es an der Entwicklung von XPS für die Automobilindustrie mitzuarbeiten. Die Firma INPRO ist eine in Berlin ansässige Firma deren Gesellschafter sich aus der Automobilindustrie und deren Zulieferer zusammensetzen. Sie arbeitet ausschließlich für die Gesellschafter und nicht für den Markt.

In den acht Jahren von 1985 bis zu meinem Ausscheiden im Jahre 1993 arbeitete ich als Systementwickler, Knowledge Engineer und Projektleiter am Aufbau der Expertensystem-Shells PRIMUS (Transferstraßendiagnostik einer Grob 4044 im VW-Werk Braunschweig) und SIGMA (Transferstraßen-Diagnostik einer Excel-O Transferstraße des Daimler-Benz Werks Marienfelde, Berlin) und mehreren Wissensbasen, die mit Hilfe dieser Shells erstellt wurden.

1985 waren wir als Expertensystementwickler in einer Situation, die man sich heute schwer vorstellen kann. Wir kauften zum Beispiel den ersten AT-Computer, den es auf dem Markt gab, zur Entwicklung der Shell PRIMUS. Die Ausstattung des Rechners auf mehrere Megabyte kostete auch mehrere 1000 DM. Wegen betriebsinterner Regelung bei VW durfte ein Bandarbeiter keine Tastatur bedienen. Die Ausgabe musste über die Funktionstasten realisiert werden. Die Entwicklungssprache der Expertensysteme war das langsame LISP. Um den Dialog schneller zu machen wurde die Dialogkomponente in Assembler geschrieben. In meiner Doktorarbeit hatte ich mit verschiedenen Assemblern gearbeitet, so dass ich diese Aufgabe übernahm.

Da die angebotenen Softwareinterrupts langsam waren, griff ich direkt auf den Bildschirmspeicher zu. Dabei kam es folgende Begebenheit. Beim Experimentieren mit dem Bildschirmspeicher gab ich einen Punkt auf dem Bildschirm aus. Ich sitze also vor einem schwarzen Bildschirm mit einem

Punkt und jubele vor Freude, was sowohl beim Chef wie bei die Kollegen nur leichtes Kopfschütteln erzeugte.
Die Entwicklung einer Expertensystem-Shell dauert mehrere Jahre. Wir mussten dabei alles selbst entwickeln. Es gab keine Literatur über Knowledge Engineering und die vorhandenen Shells waren nicht angepasst auf unsere Problematik. Außerdem sollte zum ersten Mal ein PC im Bereich der Expertensystemtechnologie eingesetzt werden. Im Bereich der PC-Entwicklung arbeitete ich zunächst allein. Als Informatiker ist man aber in diesem Bereich des Maschinenbaus etwas überfordert. Obwohl ich als Nebenfach meines Studiums an der TU Berlin Messe- und Regelungstechnik belegte, überzeugte ich meinen Chef davon, dass ich unbedingt die Unterstützung eines Maschinenbauers brauchte. Dieses Team hat sich im Folgenden sehr bewährt. Als ich zum ersten Mal vor einer Transferstraße stand und mir die Ausdehnung, die unzähligen Stationen und deren Teile und den mir fremden Produktionsvorgang gewahr wurde, war ich für einen Fachmann an meiner Seite sehr dankbar.
Bei der Recherche für dieses Buch hatte ich oft den Eindruck, dass sich viele Menschen mit der Technologie beschäftigen, selbst aber noch kein größeres Expertensystem realisiert haben. Die Expertensysteme, die ich in meiner Zeit bei der INPRO entwickelt habe, beinhalteten mehrere 1000 Teile und mehrere 1000 Regeln. Es sei mir deshalb verziehen, dass ich manche Dinge aus der Sicht eines Pragmatikers sehe. Die Kritik, die ich an mancher Stelle anbringe, sollte unter diesem Gesichtspunkt gesehen werden.
Als ich in der Industrie arbeitete war die Expertensystem-Technologie gerade auf dem Höhepunkt ihrer Erwartung angelangt. Wie in der Künstlichen Intelligenz üblich, hatte man sehr hohe Erwartungen in die Technologie, die sich aber nicht erfüllten. Nach einem Anschwellen der Expertensysteme in den neunziger Jahren kam es dann zum so genannten KI-Winter wegen zu hoher Erwartungen /Unikassel 07//Bachelier 04//Reminger 88/.
/Cawsey 03/weist darauf hin das es allgemeine Punkte gibt, die für den Entwurf aller Systeme relevant sind, er nennt dabei

- die Art des Problems,
- die Art wie das System normalerweise entwickelt wird und
- die Gesamt-Architektur, die verwendet wird.

Wenn er anmerkt, dass die Entwicklung von Expertensystemen meist sehr teuer ist, spricht er mir dabei aus der Seele. An der Entwicklung einer der oben genannten Shell waren bis zu sieben Menschen mehrere Jahre beschäftigt.
/ Cawsey 03/ weist zu Recht darauf hin, dass eine realistische Einschätzung des Nutzens eines Expertensystems oft wegen des Enthusiasmuses der Entwickler

unterbleibt. Der Vorteil bei meinem Arbeitgeber war es, dass er nicht für den freien Markt arbeitete sondern nur für die Gesellschafter. In einer solchen Konstellation war es möglich auch neue Technologie auszuprobieren. Die entstandenen Expertensystems-Shells waren unterschiedlich erfolgreich. Wir brauchten einige Zeit bis wir versiert genug waren

Das vorliegende Buch stellt eine Einführung in die Expertensystemtechnologie dar. Es kann nicht dem neuesten Stand der Technik enthalten, sondern wendet sich an Studierende, die sich mit der Problematik zum ersten Mal beschäftigen. Ich weise im Buch immer wieder auf andere Autoren hin, die weiterführende Literatur erstellt haben. Es war mir auch nicht wichtig irgendeiner der wissenschaftlichen Thesen zu folgen, etwa dass die Expertensystem-Entwicklung nur mit bestimmten Tools - etwa KADS - vorgenommen werden sollte.

Das Buch entstand aus einer Vorlesung heraus, die ich im Masterstudiengang *Informatik und Kommunikationssysteme* für den Schwerpunkt Künstliche Intelligenz halte. In dieser Vorlesung werden ein Expertensystem und eine Wissensbasis entwickelt. Natürlich geht die Entwicklung eines Expertensystems-Shells innerhalb einer solchen Vorlesung langsam vonstatten. Das Buch zeigt keine Implementierungsdetails auf, sondern will die Grundlagen legen. Der Stand der Expertensystem Shell kann auf meiner Homepage erfragt werden.

Ich entwickele mit meinen Studenten innerhalb der Vorlesung eine eigene Wissensrepräsentations-Sprache, die wir dann zum Aufbau der Wissensbasis verwenden. Im Kapitel 5 (das rote Wunder) wird sowohl die Wissensrepräsentation-Sprache wie einige Wissensbasen vorgestellt. Wir verwenden dabei ein in der Hochschule vorhandenes Modell einer Transferstraße, die der Kollegen der Prozess-Daten-Verarbeitung auch weiter ausbauen möchte.

Das Buch kann grob eingeteilt werden in die Bereiche:

- Aufbau und Funktionsweise eines Expertensystems, indem auch ein kleinerer Abschnitt der Geschichte der Expertensystem vorhanden ist,
- Wissenserwerb, indem ich verschiedene Methoden des Knowledge Engineering vorstelle und
- dem Aufbau von Wissensbasen anhand der Modell-Transferstraße.

Dieses Buch basiert auf den Erfahrungen meiner industriellen Tätigkeit, meinen Arbeiten im Bereich des Einsatzes von wissensbasierten Systemen im Bereich der lokalen Netze und den daraus entwickelten Vorlesungen.

2 Grundlagen

Wenn man sich längere Zeit mit Intelligenz beschäftigt, ist man manches Mal von den erstaunlichen Fähigkeiten einer Stubenfliege fasziniert. Könnte man ein solches Wunderwerk nur bauen!

2.1 Geschichtliche Einordnung

2.1.1 Der Begriff „Künstliche Intelligenz“

Der Begriff Künstliche Intelligenz (KI) ist eine Übertragung des englischen Begriffs *Artificial Intelligence* ins Deutsche. Die wörtliche Übersetzung aus dem Englischen ist dabei irreführend. Artificial sollte nach meiner Meinung vielmehr als "unecht" oder "Schein-", der Begriff Intelligenz als "Verständnis" übersetzt werden.
Künstliche Intelligenz erschafft damit ein Scheinverständnis, eine unechte Intelligenz. Systeme (nicht nur Computerprogramme), die sich scheinbar intelligent verhalten, die scheinbar ein Verständnis besitzen, sollen mit Hilfe von Methoden dieses Gebiet realisiert werden /TURING 87/. Nach meinem Verständnis ist dabei nicht die Entwicklung anderer Intelligenzformen das Ziel der daran beteiligten Wissenschaftler, sondern von Systemen, die sich scheinbar intelligent verhalten. Der versierte Leser merkt, dass ich hier die „schwache“ KI propagiere. /wiki 10g/
Leider besitzen Probleme, die Intelligenz erfordern, meist sehr hohe Komplexität und lassen sich nicht unmittelbar durch überschaubare und leicht einsichtige Algorithmen lösen. Damit wir verständige Systeme realisieren können, müssen wir diese erst selbst verstehen. Nur ein Tor glaubt er hätte die Welt verstanden./ Dreyfus 79// Feigenbaum 84// Hartmann 01/

2.1.2 Turing vs Searle

Der geniale Mathematiker Alan Turing näherte sich im Jahre 1950 in seinem aufsehenerregenden Aufsatz "Intelligent Machinery" dem Begriff der maschinellen Intelligenz. Er erklärte: "Ich möchte mich mit der Frage beschäftigen, ob es Maschinen möglich ist, intelligentes Verhalten zu zeigen."/TURING 87, S. 83/
Wie konnte aber festgestellt werden, ob eine Maschine intelligentes Verhalten zeigt? Turing führte zunächst das Imitationsspiel ein. In diesem Party-Spiel versuchen zwei Personen einen Schiedsrichter davon zu überzeugen, dass sie beide Mann oder Frau sind, obwohl sie unterschiedlichen Geschlechtern angehören. Der Schiedsrichter kann die Personen weder sehen noch hören und muss anhand von schriftlich gestellten und beantworteten Fragen ermitteln, welche Person das falsche Geschlecht angibt. Dabei wird er aber den Lügner nur in einer bestimmten Anzahl von Fällen erraten können.

Abbildung 2.1: *Imitationsspiel*

Im Turing-Tests "ob eine Maschine intelligentes Verhalten zeigt", ersetzt man nun eine der Personen durch die Maschine bzw. durch ein Programm. Der Schiedsrichter soll nun ermitteln wer Mensch und wer Maschine ist.
Eine Maschine zeigt dann intelligentes Verhalten, wenn sich der menschliche Schiedsrichter beim Test dieser beiden Parteien prozentual nur genauso oft irrt wie der Schiedsrichter des Imitationsspiels.

Abbildung 2.2: *Turing-Test*

Die Annahme von Turing, dass ein Computer diesen Test bis zum Jahr 2000 bestehen würde, erfüllte sich allerdings nicht. Trotz aller Fortschritte in der Computertechnologie, trotz der Tatsache, dass man dem Computer das Schreiben von Gedichten und selbst die Ungeduld oder das "verärgert sein" beibrachte, konnte bisher kein Computer den allgemeinen Turing-Test bestehen. Das Backgroundwissen eines Menschen über die Welt ist einfach zu groß, womit die Anzahl der möglichen Fragen, die man über die Welt stellen kann den Computer immer noch überfordert. Schränkt man den Wissensbereich stark ein, z.B. auf mathematisches Wissen hat das Programm hingegen größere Chancen etwas „vorzuspielen".
Die Probleme des Turing-Tests lassen sich mit der folgenden kleinen Geschichte verdeutlichen. Innerhalb einer Testreihe wurden Studenten als Schiedsrichter ausgewählt. Sie sollten über zwei Computer Fragen stellen. Ein Computer war mit einem Menschen verbunden, auf dem anderen lief ein Programm. Die Sache verlief in den gewohnten Bahnen, bis ein Student sich auf den Stuhl setzte und nichts tat. "Sie dürfen machen, was sie wollen", versucht ihn der Versuchsleiter zu motivieren."Sie dürfen jede Frage stellen, die Ihnen einfällt." "Ich will gar nichts machen!" antwortet ihm der Student. Nach einer viertel Stunde erscheint auf dem einen Bildschirm "Hat der Test schon angefangen?" Der Student deutete auf den Bildschirm und erriet: "Das ist der Mensch!"
In den 1980er Jahren zeigte John Searle (geb. 1932) mit einem Gedankenexperiment die Schwächen des Turing-Tests auf. Das als „Chinesischen Zimmer" bekannte Experiment sollte zeigen, dass wir niemals beweisen können ob ein Computer - ungeachtet seines Verhaltens - Verstand

oder Verständnis besitzt. Die folgende Beschreibung des Searlechen Experiments erfolgt nach. /Dabringer 09/

Beim Chinesischen Zimmer – Experiment befindet sich ein Mensch in einem geschlossenen Raum. Dieser Raum besitzt einen Schlitz, über den dem Menschen Zettel zugesteckt werden können. Auf diesen Zetteln steht in chinesischer Notation eine Geschichte und Fragen zu dieser Geschichte. Der Mensch selbst ist des Chinesischen nicht mächtig. Somit kann er weder die Geschichte noch die Fragen verstehen. Er findet in diesem Raum einen Stapel chinesischer Skripte und Schriftzeichen und eine Anleitung in seiner Muttersprache vor. Die Anleitung zeigt ihm genau auf, was er tun soll wenn auf einem hereingereichten Zettel ein bestimmtes Schriftzeichen auftaucht.
Die Zuordnung der Geschichte zum Skript und das „Verständnis" einer entsprechenden Frage erfolgt allein durch einen Vergleich der Form der verwendeten Schriftzeichen. Die Anleitung schreibt dem Menschen nun vor, welche Zeichen er nach dem Erkennen eines bestimmten Zeichens auf den Antwortzettel zu übertragen hat.
Vor der Tür nimmt ein chinesischer Muttersprachler die Antwortzettel in Empfang und kommt zu dem Schluss, dass sich ein Chinesisch sprechender Mensch im Raum befindet.

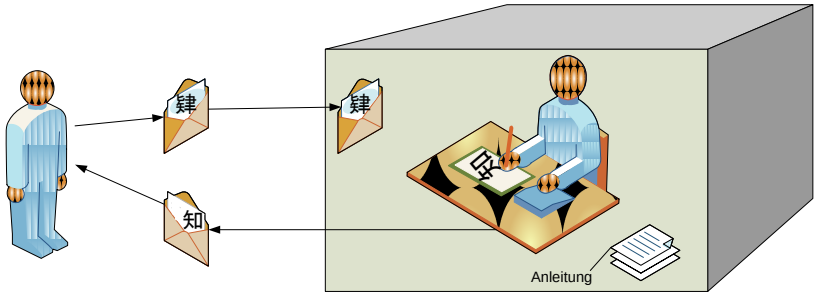


Abbildung 2.3: *Chinesisches Zimmer*

2.1.3 Summerschool for Artificial Intelligence

Ausgehend von den von Turing aufgeworfen Fragen beschäftigten sich verschiedenen Forschern völlig unterschiedlicher Disziplinen mit der Abbildung natürlicher Fähigkeiten auf einen Computer. John McCarthy, Marvin Minsky, Nathan Rochester und Claude Shannon organisierten eine Konferenz, die sich mit den entsprechenden Fragestellungen befasste und am 13. Juli 1956 am Dartmouth College in Hanover, New Hamsphire begann. McCarthy prägte den Begriff „artificial intelligence“ („künstliche Intelligenz“) bereits1955 in dem Förderantrag für diese Konferenz./wiki 10g/
"Dieses Forschungsprogramm basiert auf der Annahme, dass jeder Aspekt von Lernen sowie jedes andere Kennzeichnen von Intelligenz im Prinzip so präzise beschrieben werden können, dass eine Maschine sie simulieren kann." / Mc Corduck 79, S. 93, Gardner 89, S. 153/
Wie andere Forscher vorher hatten die KI-ler dann schnell die Erwartung, sehr große Fortschritte in kurzer Zeit machen zu können. Es wurden angenommen, dass innerhalb der nächsten zehn Jahre ein Computer Schachweltmeister sei (man hatte gerade Schachprobleme mit einem 4 x 4 Brett gelöst), ein Computer einen wichtigen mathematischen Satz finden oder beweisen werde oder man1966 keinen Übersetzer mehr bräuchte./wiki 10g/
Als Beispiel eines mechanisierten Übersetzungsvorganges der damaligen Zeit wird oft der Satz "der Geist ist willig, aber das Fleisch ist schwach" genannt, der nach Übersetzung und Rückübersetzung vom Englischen ins Russische zu "der Wodka ist gut, aber das Steak ist schlecht" verwandelt wurde.
Wie dieses Beispiel zeigt wird die Komplexität von - für uns als gewöhnlich erscheinenden - Handlungen meist weit unterschätzt. Im Allgemeinen gleicht das Ringen um Erkenntnis eher der Arbeit des Sisyphos als der des Cäsars. So brauchte es 47 Jahre bis es dem von IBM entwickelten System Deep Blue 1997 gelang den Schach-Weltmeister Garry Kasparov in sechs Partien zu schlagen./wiki 10g/
Die KI-Forscher wandten sich in den 1960er zunächst einer universellen Problemlösung zu. Newell und Simon wollten mit dem General Problem Solver ein Programm entwickeln, das mit einfachen Methoden beliebige Probleme lösen können sollte. /wikigps 10/
Nach dem Scheitern dieses etwas zu ambitionierten Forschungsprogramms wendete man sich zunächst allgemeinen, effizienten Suchstrategien zur Lösung klar formulierter Probleme etwa dem Beweis mathematischer Sätze oder Schachprobleme zu. So entstanden etwa der A* Algorithmus /Schmidt 10/,

die Alpha-Beta Suchstrategie für Spiele (Schach, Dame, Mühle) /wiki 10a/ und die Anwendung der Resolutionsmethode /wiki 10e/ in der KI.
Welche bisweilen naiven Vorstellungen die Forscher der Anfangsjahre hatten zeigt sich am Beispiel von McCarthy, der nach /wiki 10g/ 1958 vorschlug, das **gesamte** menschliche Wissen in eine formale Darstellungsform, etwa die Prädikatenlogik 1. Stufe, zu bringen. Dies klingt sehr nach dem Pascalschen Traums, nach dem man - hätte man erst alles formalisiert - nur noch rechnen bräuchte, um zu wissen was die Wahrheit ist.
Wie schnell einem Menschen die Illusion eines beseelten Partners vermittelt werden konnte zeigte sich durch das Ende der 1960er Jahre von Weizenbaum entwickelte simple Programm ELIZA, in dem der Dialog eines Psychiaters mit einem Patienten simuliert wird. /wiki 10g/

2.1.4 Unterteilung der KI

Basierend auf den Arbeiten von Alan Turing formulierten Allen Newell und Herbert Simon die *Physical Symbol System Hypothesis*, nach der Denken Informationsverarbeitung ist, Informationsverarbeitung ein Rechenvorgang also Symbolmanipulation ist und es auf das Gehirn als solches beim Denken nicht ankommt: Intelligence is mind implemented by any patternable kind of matter. /wiki 10g/
Moravec und Kurzweil definieren mit dem **artificial brain argument** die These, dass das menschliche Gehirn simulierbar sei und nach der Übertragung in Hardware und Software diese mit dem Original im Wesentlichen identisch sei. /Dabringer 09/
Diese beiden Thesen umschreiben den harten KI-Ansatz, während die Vertreter der schwachen KI keine künstlichen Wesen schaffen wollen, sondern an der Abbildung intelligenter Fähigkeiten auf dem Rechner arbeiten, dabei aber keine Evolution des Menschen zum Roboter im Sinne haben. /wiki 10g/
Eine andere Betrachtungsweise ordnet die Methoden der KI grob in zwei Dimensionen ein: Symbolische vs. Neuronale Künstliche Intelligenz und Simulationsmethode vs. Phänomenologische Methode.
Die Neuronale Künstliche Intelligenz möchte dabei das menschliche Gehirn möglichst präzise nachbilden. Die Symbolische Künstliche Intelligenz nähert sich den Intelligenzleistungen von einer begrifflichen Ebene her. Die Simulationsmethode orientiert sich so nah wie möglich an den tatsächlichen

kognitiven Prozessen des Menschen während es dem phänomenologischen Ansatz vor allem auf das Ergebnis ankommt. /Dabringer 09/

Zeitlich lässt sich die Geschichte der KI schliesslich in eine Klassische Periode (1955 -1965), eine romantische Periode (1965 -1975) und eine moderne Periode (ab 1975) einteilen. /Kurbel 92/
Die Künstliche Intelligenz ist heute in sehr viel verschiedene Teilgebiete aufgeteilt, die hier nicht im Einzelnen und in ihrer geschichtlichen Entwicklung vorgestellt werden können. Ich verweise hier auf *the Handbook of Artificial Intelligence I-III* /Barr 81-82/.

2.1.5 Teilgebiete der Künstlichen Intelligenz

Die folgende Argumentation entnehmen ich weitgehend meinem Beitrag in /Hartmann 01/, in dem ich die verschiedenen Thesen noch weiter ausführe.
Die Künstliche Intelligenz lässt sich in verschiedene Teilgebiete und ihre Methoden unterteilen / BAR81-82/. Ohne auf die einzelnen Gebiete näher eingehen zu wollen, werden dabei sowohl geistige wie auch körperliche Fähigkeiten abgebildet. Der geneigte Leser sei darauf hingewiesen, dass diese Einteilung nur eine von verschiedenen ist. So wurde der Theorem Beweiser hier nicht beachtet. Die einzelnen Gebiete werden immer wieder neu angeordnet, bleiben dabei aber auch immer, wenn auch bis weil unter anderem Namen erhalten. So kann man die Neuronalen Netze sowohl als eigenständiges Gebiet (wegen ihrer Bedeutung und der Größe des Gebiets), wie als Repräsentationsform verstehen. /Hartmann 01/

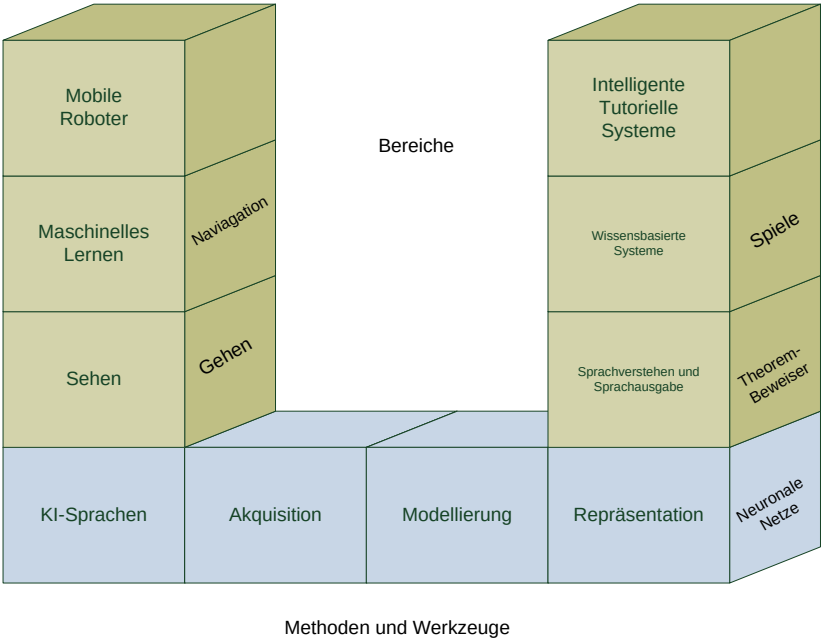


Abbildung 2.4: *Bereiche der Künstlichen Intelligenz (Methoden blau)*

Die im Weiteren als "körperliche Intelligenz" von mir bezeichneten Fähigkeiten stellen in der Künstliche Intelligenz Nachbildung von körpereigenen Teilen oder von körperlichen Eigenschaften von Tieren und Menschen dar, also beispielsweise die Abbildung des vegetativen Nervensystems oder die Nachbildung einer Hand. Als "geistige Intelligenz" verstehe ich die Abbildung von Fähigkeiten des Gehirns oder kognitiver Fähigkeiten. Damit ergibt sich als Folgerung:
da alle Lebewesen sowohl körperliche wie auch geistige Fähigkeiten besitzen, besitzen alle Lebewesen ob Baum, Amöbe oder Mensch Intelligenz. Diese ist nur verschieden ausgeprägt.
Der Grad der gesamten Intelligenz ist proportional der Menge von Fähigkeiten.

Eine Wertigkeit der Fähigkeiten wird ganz bewusst nicht vorgenommen. Der Grund sei mit folgendem Witz verdeutlicht.
Ein praktischer Physiker, ein theoretischer Physiker und ein Mathematiker werden mit einer geschlossenen Dose Fisch in einem leeren Raum mit kahlen Wänden eingesperrt. Sie haben nichts anderes zu essen und keinen Dosenöffner. Nach einer Woche öffnet man die Räume und sieht nach den Eingeschlossenen.
Erster Raum - der praktische Physiker: Die Wände sind voller Dellen. Der Physik sitzt vor der geöffneten Dose und sieht sehr zufrieden aus.
Zweiter Raum - der theoretische Physiker: Die Wände voller Formeln, eine Delle. Auch er sieht gesättigt aus.
Als man den dritten Raum betritt sieht man, wie ein völlig entkräfteter Mathematiker über die Dose gebeugt die Worte murmelt: "Angenommen die Dose sei offen!"
Alle Mathematiker mögen es mir verzeihen, solche Scherze gibt es auch über Informatiker.
Alle Fähigkeiten des hochspezialisierten Mannes waren in dieser Situation nutzlos. Ähnlich würde es wohl einem gewöhnlichen Mitteleuropäer ohne Hilfsmittel in der Wüste oder im Dschungel ergehen.

2.2 Wissensbasierte Systeme

Das Teilgebiet der Wissensbasierten Systeme (WBS) stellt den Ausgangspunkt für die Expertensysteme dar, er ist deshalb zunächst Thema der Darstellung.
Generell sollen Wissensbasierte Systeme Wissen auf dem Rechner verfügbar machen. Eine Diskussion was Wissen überhaupt ist erfolgt in Kapitel 3. Für die folgende Betrachtungsweise genügt es sich das Wissen als Verständnis der Welt, das unserem Handeln zugrundeliegt, vorzustellen. Wissensbasierte Systeme wollen damit Weltverständnis erzeugen. WBS möchten dieses Wissens zur Verfügung stellen und da es Wissens in sehr unterschiedlicher Form gibt, suchte man dazu Strukturen, die man für verschiedene Anwendungen verwenden konnte.
In herkömmlichen Programmen ist das Wissen im Programmcode selbst eingebettet. „… die Programme enthalten sowohl das Wissen über das Anwendungsgebiet – etwa über die Produktionsplanung und –Steuerung – als auch das Wissen darüber, wie mit dem ersteren umzugehen ist“ /**Kurbel 92, S. 18**/. Darüber hinaus wird auch ein allgemeines Problemlösungswissen - etwas

wie man den Algorithmus in korrekte Einzelschritte aufteilt - im Programmcode enthalten sein. Wissen über das Anwendungsgebiet ist im Programmcode starr eingebunden.
Sind Änderungen erforderlich führt dies oft zu umfangreiche Modifizierungen des Programmcodes wodurch eine niedrige Flexibilität gegenüber Veränderungen des Wissens und zusätzlich eine starke Abhängigkeit von den Programmierern des Systems besteht /Wiegand 04/.
Innerhalb des Bereichs der Wissensbasierten Systeme trennt man nun das Wissen über einen Anwendungsbereich von der Verarbeitungsstrategie des Wissens. Das Wissen zur Problemlösung ist somit nicht mehr implizit in Form fester Programm-Verarbeitungsschritte, sondern explizit in Form einer Wissensbasis und eines darauf arbeitende Verarbeitungssalgorithmusses abgebildet. /Wagner 08/
Damit erfolgt eine Trennung zwischen anwendungsbezogenem Wissen und anwendungsunabhängigen Wissen /**Kurbel 92**/. Das anwendungsunabhängige Verarbeitungswissen kann damit prinzipiell in verschiedenen Anwendungen verwendet werden.
Kurbel definiert daraus folgend:
„Ein wissensbasiertes System ist ein Softwaresystem, bei dem das Fachwissen über ein Anwendungsgebiet („Domain knowledge“) explizit und unabhängig vom allgemeinen Problemlösungswissen dargestellt wird.“ /Kurbel 92, S. 18/
Vereinfacht ausgedrückt ist ein Wissensbasierte Systeme also ein Software-Systeme, das sich durch eine konsequente Trennung zwischen Wissen und Wissensverarbeitung auszeichnen /Richter 10/.
In der Wissensbasis wird das Domain-Wissen in Form einer geeigneten Repräsentationsform dargestellt. Diese muss dem Verarbeitungsalgorithmus bekannt sein, damit er dieses Wissen verarbeiten kann. Der Verarbeitungsalgorithmus (Inferenzmechanismus, Problemlöser) kann jedes Wissen verarbeiten, dass in der entsprechenden Repräsentationsform dargestellt ist. Dabei muss man zwischen der Form, wie Wissen intern dargestellt wird und wie dieses Wissen extern beschrieben werden soll (Wissens-Repräsentations-Sprache WRS) unterscheiden. Die Repräsentationsform ist dabei etwa eine Kombination zwischen Regeln und Objekten, die Beschreibungssprache besitzt eher die Form einer Programmiersprache (siehe hierzu Kapitel 5 und /Hartmann 09/).

Beim Anweisungsbasierter System (herkömmliches Programm), das aus einer Sequenz von Befehlen und Abfragen besteht, legt der Programmierer fest, was wie und in welcher Reihenfolge es gemacht wird. Beim Wissensbasierten

System, das aus einer Menge von Repräsentationselementen besteht, legt der Ersteller der Wissensbasis fest, was getan wird. Wie und in welcher Reihenfolge bestimmt der Verarbeitungsalgorithmus. Dem Ersteller der Wissensbasis wird damit eine zentrale Rolle zugestanden./Richter 10/

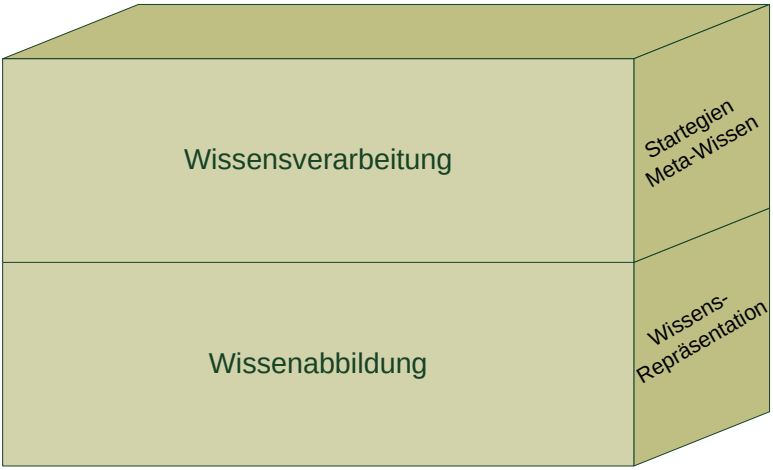


Abbildung 2.5: *Kern eines wissensbasierten Systems*

Da die Verarbeitung anhand von explizit dargestelltem Wissen erfolgt, dass durch die WRS in einer verstehbaren Form vorliegt, ist es - anders als beim herkömmlichen Anweisungsbasierten System -auch möglich Erklärungen über das Wissen zu erzeugen, etwa wie der Verarbeitungsmechanismus vorgegangen ist.
Wissensbasierte Systeme werden vor allem bei sehr komplexen oder noch nicht vollständig verstandenen Problemen eingesetzt, weil man dabei einen Fachexperten leichter einbeziehen kann, der über Prorammiertechniken keine Kenntnisse zu haben braucht. Ein solches System muss in enger Zusammenarbeit mit dem Knowledge Engineer (Spezialist der Wissensrepräsentation und Entwickler der Wissensbasis) und dem Fachexperten erstellt werden. /Wagner 08/
Aufgrund der Komplexität des Problems kann aber nicht davon ausgegangen werden, dass immer die optimale Lösung gefunden wird. Die verwendeten

Heuristiken können dabei häufig nur eine Teillösung erreichen. Anders als herkömmliche Programme, die große Mengen von Daten mit einem einheitlichen Algorithmus verarbeiten können, sind WBS immer nur für relativ überschaubare Systeme geeignet. /Wagner 08/

2.3 Vom WBS zum XPS

Während WBS einfach nur Wissen in Systemen abbilden sollen, wird von Expertensystemen erwartet, dass ganz spezielles Wissen abgebildet wird. In XPS wird Expertenwissen abgebildet. Dieses Wissen zeichnet sich dadurch aus, dass es nicht nur aus Büchern oder anderen frei zur Verfügung stehenden Quellen stammt, sondern auch Wissen umfasst, welches

- aus Erfahrungen besteht (Daumenregeln),
- mittels Assoziationen oder Analogien abgeleitet wurde und
- mittels Heuristiken ermittelt werden kann.

Erfahrungswissen wird einem Experten vermittelt, wenn er sich mit einem bestimmten Wissensgebiet längere Zeit beschäftigt. Das Wissen eines Experten geht dabei über pures Bücherwissen hinaus, weil nicht alle Erfahrungen auch dokumentiert sind. Experten bauen Wissen auf, dokumentieren diese aber nicht unbedingt, weil sie selbst keine geeignet Form kennen oder weil sie nicht vermuten, dass diese Aufschreibung etwas nutzt.
Ein Experte kann zwischen bestimmten Ausgangszuständen auf bestimmte Endzustände schließen, ohne dass es zwischen Anfangs- und Endzustand eine Kausalkette geben muss bzw. diese bekannt ist (Assoziation). In der Analogie stellt er Ähnlichkeiten zu einem bereits bekannten Fall fest, der bereits dokumentiert ist. Diese Analogien und Assoziationen können dabei weitgehend nur durch den Experten gefunden werden und ihr Zustandekommen ist zum Teil auch nicht von ihm detailliert beschreibbar.
Heuristische KI-Verfahren kommen vor allem in der Verarbeitungsstrategie des Wissens zum Einsatz. Sie erleichtern die Wissensverarbeitung, wenn komplexe Strukturen durchsucht werden müssen oder die Verarbeitung gesteuert werden soll.
Ein XPS stellt durch die Auswahl dieses besonderen Wissensbereichs eine Teilmenge der Wissensbasierten Systeme dar. Ich schließe mich damit der Argumentation aus /Kurbel 92/ an, die auf Watermann zurückgeht, allerdings nicht unumstritten ist.

Danach stellen KI-Programme scheinbar intelligentes Verhalten dar. Wissensbasierte Systeme sind eine Teilmenge davon, in denen das Wissen und Wissensverarbeitung voneinander getrennt vorliegen. Für die Darstellung und Auswertung des Wissens werden vorwiegend Repräsentationsformen und Methoden der KI verwendet. Watermann versteht XPS nun als Teilmenge der WBS, da sie Expertenwissen abbilden. Dieses Wissen bezieht sich auf ein schwerwiegendes Problem der realen Welt. Anzumerken sei dabei noch, dass XPS nur dort sinnvoll sind, wenn es entsprechende Experten gibt.

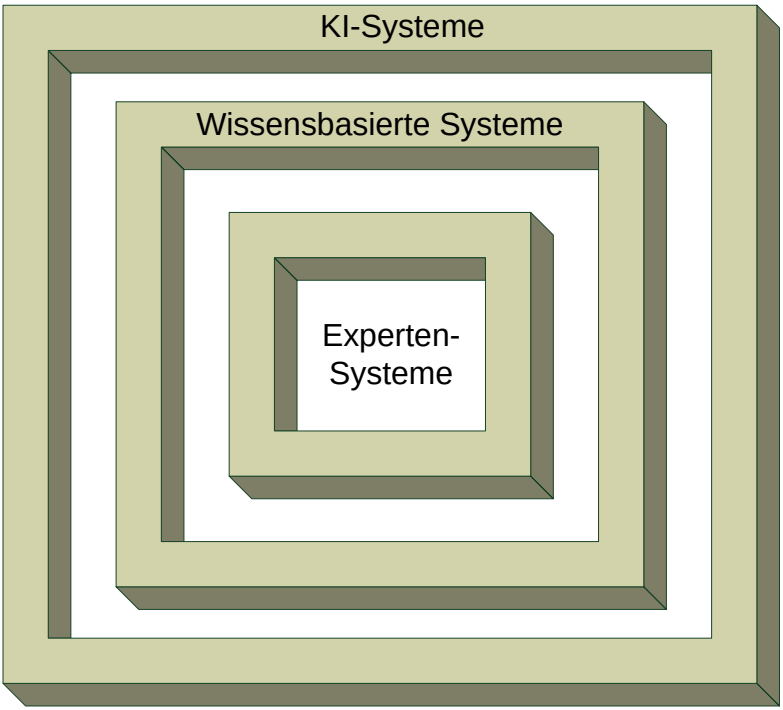


Abbildung 2.6: *Zusammenhang zwischen WBS und Expertensystemen*

2.3.1 Definition eines Expertensystems

/Kurbel 92, S. 22/ definiert ein XPS als „ein Programm, das in einem eng begrenzten Anwendungsbereich die spezifische Problemlösungsfähigkeiten eines menschlichen Experten zumindest annähernd erreicht oder übertrifft."
In Anwendungskreisen werden XPS und WBS oft gleichgesetzt. /Kurbel 92, S.26/ weist aber darauf hin, dass „Wenngleich XPS meist wissensbasiert konstruiert werden, so liegen die Begriffe doch auf unterschiedlichen Ebenen."
Die Bezeichnung XPS beschreibt das Verhalten des Systems gegenüber seiner Umwelt. Es verhält sich so ähnlich wie einem menschlichen Experten. Wissensbasiert beschreibt aber die interne Systemarchitektur, also die Trennung zwischen Verarbeitungs- und anwendungsspezifischen Wissens.
Da nach der oben verwendeten Definition XPS nicht zwangsläufig wissensbasiert aufgebaut sein müssen, genügt es beim Benutzer den Eindruck zu hinterlassen, dass sie sich ähnlich wie ein menschlicher Experte verhalten. Bei den nachfolgenden Betrachtungen werde ich mich aber auf XPS als Teilmenge der WBS beschränken.
Auch weitere Definitionen lassen sich in der Literatur finden.
So beschreibt /Schneider 10, S.8/ ein XPS als eine System, dass „„Wissen" über ein spezielles Gebiet ansammelt und speichert, aus dem Wissen Schlussfolgerungen zieht und zu konkreten Problemen des Gebietes Lösungen anbietet."
/Wachsmuth 94, S. 104 (2)/ sieht hingegen in einem XPS „ein Computerprogramm, mit dem versucht wird, ein technisches Mittel für die Bearbeitung von Fachaufgaben zu konstruieren, welche bislang hochspezialisierten Experten vorbehalten sind."
Weil man nach dieser Definition einen menschlichen Experten i n seiner Problemlösungsfähigkeit abbilden soll, ist es sinnvoll sich zuerst zu vergegenwärtigen, welche Eigenschaften ein Experte überhaupt hat, d.h. was ihn zum Experten macht.
Nach R. Davis /Kurbel 92/ ist ein Experte in der Lage:
- Probleme zu lösen,
- Ergebnisse zu erklären,
- hinzuzulernen,
- Wissen zu reorganisieren,
- Regeln zu übertreten,
- die eigene Kompetenz zu beurteilen und
- sich mit Anstand zurückzuziehen.

Aus meiner Erfahrung im Bereich des Knowledge Engineering möchte ich hierzu die folgenden Anmerkungen machen:

a) Probleme lösen

Experten in der Industrie werden danach bewertet, wie erfolgreich sie bei der Problemlösung sind. In der Industrie kommt es nicht unbedingt auf die Schönheit und Effizienz einer Problemlösung an, sie muss vor allem schnell gehen, weil Zeit hier Geld bedeutet. Daher sind Problemlösungen, auf die man im KE stößt nicht immer elegant, sondern entsprechen der Praxis. Innerhalb der Aufnahme und Abbildung des Wissens kommen Experten oft zu neuen Problemlösungen, die dann effizienter sind.

b) Ergebnisse zu erklären

Da die schnelle Problemlösung gefragt ist, werden auch Erfahrungen verwendet, die selbst der Experte nicht erklären kann. Der Leser mache sich dies vielleicht an Beispiel des Krawatten -Bindens oder des Kuppelns beim Fahren klar. Wenn Sie einen Menschen fragen wie er eine Krawatte bindet, kann er dies meist nicht erklären sondern nur zeigen. Andere Vorgänge, etwa das Kuppeln ist so verinnerlicht, dass wir über den Vorgang selbst nicht mehr nachdenken. Genau so geht es aber einem Experten, wenn er sein Wissen anwendet. Eine Erklärung kann für viele Vorgänge nur in Zusammenarbeit mit einem KE erfolgen, der den Experten innerhalb dieser Erklärung führt.
Ein XPS kann, da es entsprechende Präsentationsformen und Komponenten besitzt sein Vorgehen auch detailliert erklären. Im Bereich der Ausbildung ist dies sogar ein Vorteil gegenüber einem menschlichen Experten

c) Hinzuzulernen und Wissen zu reorganisieren

Diese Fähigkeit empfinde ich als die wichtigste, die der Experte überhaupt haben kann, denn bei den typischen Anwendungsgebieten treffen wir auf hohe Komplexität, die es notwendig macht das vorhandene Wissen beständig zu überprüfen und neu zu organisieren. Innerhalb der technischen Diagnostik lernte der Experte ständig, weil sich die zu diagnostizierende Anlage beständig während ihrer Lebensdauer verändert. So werden zum Beispiel andere, effizientere Bauteile eingesetzt oder die Struktur der Anlage verändert, weil andere Teile gefertigt werden müssen usw. Jedes Dazulernen und jede

Veränderung des Wissens macht natürlich auch eine Reorganisation des Wissens notwendig, um dieses auch optimal einsetzen zu können.
Daher ist es notwendig, die Wissensbasen beständig zu warten und die Veränderungen nachzuvollziehen. Dies bedingt leider, dass ein XPS im Laufe der Zeit überaltert, wenn es nicht gewartet wird. Die Einführung eines Systems beendet also noch nicht die Kosten für das System.

d) Regeln zu übertreten

Experten haben ihr Wissen meist nicht in expliziten Regeln abgebildet. Diese Vorstellung resultiert vielleicht daraus, dass man bei der Abbildung des Wissens in die XPS häufig Regeln verwendet. Wenn der Experte dies versucht, wird er sehr viele Regeln erstellen, die alle nur kleine Abweichungen voneinander haben. Die Ausnahmen sind dabei in der Menge umfangreicher als alle erstellten Regeln. Im KE verbringt man sehr lange Zeit damit, verschiedene Regeln zu erstellen und weiß dennoch, dass jede Wissensbasis eine Beschränkung hat. Um Ausnahmen und kleine Unterschiede zwischen verschiedenen Problemlösungen einarbeiten zu können, muss man dem Experten die Möglichkeit lassen, sich bestimmte Systemzustände zu merken und später die Wissensbasis entsprechend zu überarbeiten.

e) die eigene Kompetenz zu beurteilen und sich mit Anstand zurückzuziehen

Dies sind Eigenschaften, die eher was mit dem Charakter eines Menschen zu tun haben und meiner Meinung nach nichts mit der Expertenfähigkeit zu tun haben. Ein Experte in der Industrie wird seine Stellung dieser Expertentätigkeit verdanken. Stellt er sich allzu oft in Frage wird er diesen Status nicht lange haben. Nach meiner Erfahrung ist die Beurteilung der Leistungsfähigkeit eines Menschen aus der Distanz wesentlich einfacher. Selbst wenn es eine ehrenhafte Forderung ist, scheint mir dies sogar in der akademischen Welt nur für ganz wenige Menschen zuzutreffen, wenn wir uns nur das Ego manches Professors anschauen, der ja auch nichts anderes als ein Experte ist.
In der Praxis habe ich hingegen sehr oft den Expertenstreit erlebt, der sich daraus nährt, dass man sich zu gut in dem Fachgebiet des anderen auszukennen glaubt.

2.4 Geschichte der XPS

/Kurbel 92/ teilte die Geschichte der KI in eine Klassische Periode (1955 - 1965), eine romantische Periode (1965 -1975) und eine moderne Periode (ab 1975). Aus der Sicht 1992 war diese Einschätzung vielleicht auch für die XPS zutreffend. Heute muss man die moderne Periode von XPS wohl weiter unterteilen etwas in eine Periode der industriellen Anwendung (1980 -1990), die mit einer Ernüchterung endete, einer Periode der Assistenzsysteme (1990 –…)/Berlinews 08/, die aber nahtlos übergeht in die Periode der integrierten Systeme (1990- ..) /Steinmann 93/, in der XPS nur ein Teil eines Gesamtsystems darstellen.

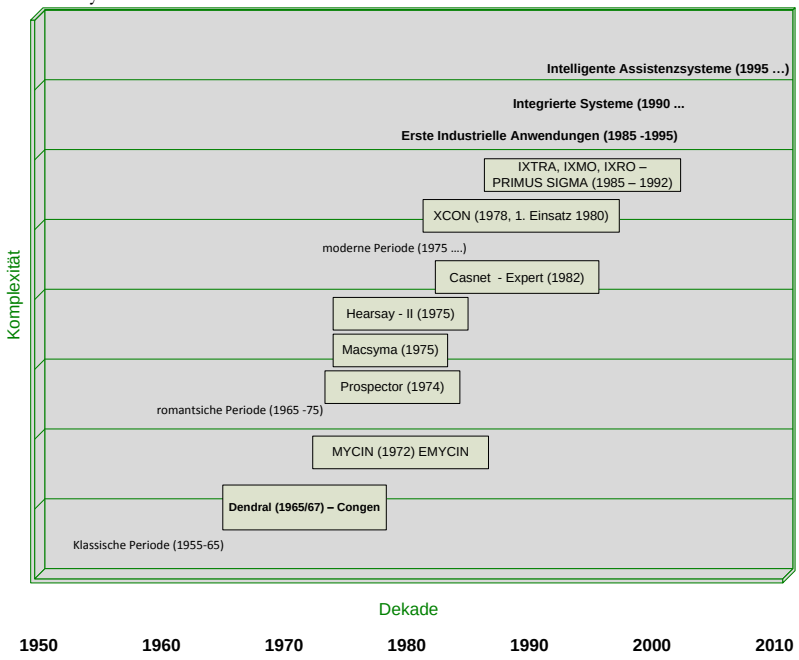


Abbildung 2.7: *Zuordnung zwischen Systemen und Perioden*

Zu Beginn der Entwicklung der KI in den 60er und 70er Jahren hatten die Forscher zu ambitionierte Ziele. Man wollte die Denkvorgänge in allgemeiner Form darstellen und versuchte dafür fundamentale Prinzipien zu erarbeiten. Nachdem die Forschungsbemühungen scheiterten schränkten die Forscher ihre Bemühungen auf das Wissen als zentraler Bestandteil des Denkvorgangs ein. Nachdem man auch hier erfolglos versuchte allgemeine Theorien über das Wissen zu erarbeiten, wandte man sich schließlich konkreten Problemen in angrenzenden Wissensgebieten zu. Damit Begann Mitte der 70er Jahren die Geschichte der Expertensysteme. /Kurbel 92/
Ende der 70er Jahre fasste E.A. Feigenbaum, die Erkenntnisse wie folgt zusammen /Kurbel 92, S. 212/:
„Das Wissen des Experten liefert den Schlüssel zum expertenmäßigen Verhalten, während die Darstellung des Wissens und die Inferenz-Strategien nur den Mechanismus zur Benutzung des Wissens darstellen. Die Suche nach einer mächtigen und ganz allgemeinen Wissensrepräsentation hat keine empirische Berechtigung. Das Expertenwissen an sich scheint sowohl notwendig als auch nahezu ausreichend für die Entwicklung eines XPS."
Damit aber wurde dem Experten selbst eine zentrale Rolle zuerkannt, denn dieser musste sein Fachwissen erst zur Verfügung stellen.

2.4.1 XPS im Verlauf der Jahrzehnte

Die im Folgenden verwendete Struktur der XPS-Systeme im Verlauf der Jahrzehnte geht auf /Steinmüller05/ und /Kurbel 92/ zurück. Kurbel geht dabei von verschiedenen Entwicklungslinien aus, die in anderer Literatur nicht so explizit dargestellt wird, aber nach meiner Meinung ihre Berechtigung hat, weil sie die Abhängigkeit der verschiedenen Systeme aufzeigt.

a) Die Dendral – Linie
Die Entwicklung des XPS Dendral /Britannica 10/, /Lederberg 87/ hat als Anwendungsfall die organische Chemie. Es wurde nach Kurbel in der Zeit ab 1967 an der Stanford-Universität entwickelt. Steinmüller setzt den Startpunkt hier auf 1965. Dendral analysiert aus der Massenspektrographie gewonnen Daten und versucht dadurch die molekulare Struktur einer vorgegebenen Substanz abzuleiten.
Im Folgesystem Meta-Dendral wird das System um Wissen darüber erweitert wie man die Analyse einer Substanz am besten durchführt (von Steinmüller als

Lernkomponente bezeichnet). Als weitere Erweiterung wird bei Steinmüller das System Congen (für Constrained Generation) erwähnt /Rehberger 95/.

b) Die Macsyma – Linie
Diese nach Kurbel mit dem System Saint /Bischoff 10/ begonnene Linie wurde am Massachusetts Institut of Technology (MIT) für den Bereich der symbolischen Mathematik ab 1975 entwickelt. Macsyma löst dabei mathematische Probleme im Bereich der Differential- und Integralrechnungen mit symbolischen Methoden./Maxima 09/
/Petti 03/ hat in einer veröffentlichten Mail die Entwicklung des Systems aus eigener Erfahrung sehr anschaulich beschrieben.

c) MYCIN und weitere medizinische Anwendungen

Ab 1972 beschritt die XPS-Technologie mit der Entwicklung von MYCIN den medizinischen Bereich. Obwohl das System nicht zum praktischen Einsatz kam, beeinflusste es den Bereich der XPS maßgeblich, weshalb es auch oft als der Großvater aller XPS bezeichnet wird. MYCIN führte die angesprochene klare Trennung von Wissensrepräsentation und Problemlösungs-Strategie zum ersten Mal durch, führte eine Erklärungs-Komponente ein und verwendete Wichtungsfaktoren (Certainty Factors, siehe Kapitel 4.6)zur Verarbeitung von vagem und ungenauem Wissen. /Britannica 10a//Beierle 08/
Mit der Entwicklung von Theresias, einer die Wissens-Akquisition unterstützenden Komponente waren die heute beim klassischen Aufbau eines XPS genannten Komponenten vollzählig. Theresias unterstützt den Benutzer beim Aufbau von neuen Wissensbasen und der Formulierung von neuen Regeln. /Heß 02/ /Spreckelsen 08/
MYCIN entstand an der Stanford-Universität. Die Entwickler E.A. Feigenbaum und E.H. Shortliffe ließen sich dabei von den Erkenntnissen, die bei Dendral gewonnen wurden, inspirieren.
Als Anwendungsgebiet wurde zunächst bakteriologische Erkrankungen des Blutes und Hirnhauterkrankungen untersucht. Später wurde das Wissensgebiet auch auf andere Infektionskrankheiten erweitert /Kurbel 92/.
Der Ablauf einer MYCIN-Sitzung wird als typisch für viele Sitzungen im Diagnostik-Bereich angesehen. Kurbel skizziert sie so /Kurbel 92, S.33/:
„
Im ersten Sitzungsteil werden die bereits bekannten Daten über einen Patienten erhoben: neben persönlichen Daten wie Name, Alter, Geschlecht etc. vor allem Ergebnisse aus bereits durchgeführten Untersuchungen und

Befunden. Mycin fragt z.B. welche Kulturen angelegt und welche Organismen dabei entdeckt wurden. Der Arzt kann stets mit „unbekannt“ antworten.
Im Weiteren werden Annahmen über mögliche Erreger generiert. Um diese Annahmen validieren zu können, erfragt das System vom Arzt weitere Informationen oder Einschätzungen. Je genauere Angaben diese machen kann, umso eindeutiger wird später die Diagnose.
Nach im Schnitt 30 bis 90 Fragen ist Mycin in der Lage, seine Diagnose zu stellen. Diese ist, wie auch beim menschlichen Experten, dem Arzt, nicht unbedingt eindeutig. Das heißt, eine Therapie muss eventuell mehrere mögliche Erreger in Betracht ziehen.
Da das Zusammenwirken mehrerer Antibiotika gefährlich sein kann, ist es nicht ratsam, jede Möglichkeit mit einem gesonderten Präparat zu behandeln. Mycin unterbreitet dem Arzt deshalb auch einen Therapievorschlag, der die Risiken minimiert.
„

Die Erklärungskomponente ermöglicht es dem Arzt dabei, dass er sich das Verhalten des Systems und die entsprechende gefundene Diagnose zu jedem Zeitpunkt erklären lassen kann.
Mit MYCIN wurde auch zum ersten Mal ein Expertensystem-Werkzeug entwickelt. Durch die Entfernung der Wissensbasis wurde aus MYCIN EMYCIN (für EmptyMycin). Diese Shell stellte nun eine problemunabhängige Komponente dar. Die Wissensbasis wird in einer dem System bekannten Beschreibungssprache definiert und kann dann von der Shell abgearbeitet werden. Dieser Ansatz wurde von anderen Entwicklungen später in Richtung Expertensystem-Toolkit erweitert. Bei einem solchen Toolkit können alle Komponenten eines XPS an den Problemlösungsfall angeglichen werden. Dies ist bei einer Shell noch nicht möglich. (siehe hierzu auch Kapitel 4.1.2)
Mit Hilfe von EMYCIN wurde beispielsweise das System Puff entwickelt, welches sich mit Lungenerkrankungen beschäftigt.

Als weitere Expertensysteme in der Medizin entstanden nach /Steinmüller05/ 1978 Casnet zur Diagnose und Therapie einer Augenerkrankung mit der daraus entwickelten Shell Expert (1982) und Internist, welches in der inneren Medizin eingesetzt wird und hier bei 75% der Krankheiten unterstützend eingesetzt werden kann. Aus Internist ist die Shell Caduceus abgeleitet.

d) Prospector (1974)

Basierend auf Erkenntnissen aus MYCIN wurde zwischen 1974 und 1983 am Stanford Research Institute (SRI) das XPS Prospector entwickelt, dass Messdaten aus Probebohrungen interpretiert und so Geologen bei der Auffindung von Lagerstätten unterstützen soll. Die Wissensbasis enthält nach Kurbel etwa 1000 Regeln.
Das System konnte nach Steinmüller seine Qualität durch den Nachweis eines Molybdän-Vorkommen (150 Mill. Dollar) beweisen.

e) Hearsay - II (1975)

Die Hearsay-Linie wurde ab 1975 an der Carnegie-Mellon-Universität in Pittsburgh (USA) gestartet. Hearsay ist ein System zur Spracherkennung und war bereits 1975 in der Lage gesprochene Sätze einfacher Grammatik im Rahmen eines Vokabulars von ca. 1000 Wörtern zu verstehen.
Im Rahmen dieses Systems wurde zum ersten Male die Blackboard-Architektur verwendet. Bei diesem Systemansatz kommunizieren mehrere kooperierende Teilsysteme über einen gemeinsam benutzten Speicherbereich, der als „Wandtafel" bezeichnet wird.
Aus Hearsay II wurde das XPS-Entwicklungswerkzeug HS III entwickelt.

f) Xcon (R1) (1978, erster Einsatz 1980) /Wiki 10i/

Das kommerziell wohl erfolgreichste XPS stellt das aus der OPS-Linie stammende Konfigurationssystem XCON (R1) dar, welches von der Firma Digital Equipment zur Konfiguration von VAX-Computern verwendet wurde. Durch das System, dass aus mehreren tausend Regeln bestehen sollte, soll die Firma jährlich Millionen von Dollar eingespart haben /Bruse 10/. (für ähnliche Systeme siehe /Hinterding 95/)

Nachdem die verschiedenen Institute Erfahrungen in der Technologie erlangten und diese auch propagiert wurden, nahm sich auch die Industrie den XPS an. Da im Zusammenhang mit XPS aber immer mit vertraulichen Daten gearbeitet wurden, die zum Teil Produktionsprozesse beschreiben, wurden viele dieser Systeme zunächst nicht publiziert.
In den Jahren 1985 bis 1992 entwickelte ich bei der Firma INPRO Diagnosesysteme im Bereich der Motorendiagnose (IXMO), der

Roboterdiagnose (IXRO) und der Transferstraßendiagnose (IXTRA), wobei mit PRIMUS und SIGMA zwei verschiedene XPS-Entwicklungssysteme (Shells) erstellt wurden./Dröll 88//INPRO 89//INPRO 90//INPRO 88//INPRO 93/
In den 90ziger Jahren des letzten Jahrhunderts kam es dann aber zu einem Stillstand in der Entwicklung, weil die Erwartungen einfach zu hochgeschraubt worden waren und man die Kosten und den Aufwand zur Erstellung und Wartung solcher Systeme sehr unterschätzt hatte /Bachelier 04/.

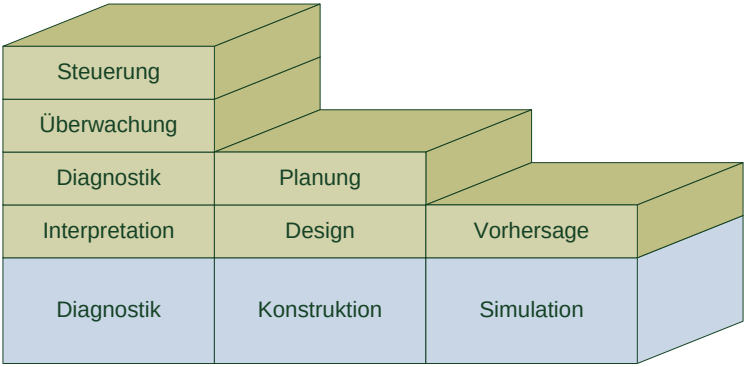
2.5 Einsatzbereiche von XPS

In diesen Abschnitt möchte ich kurz die Einsatzbereiche von XPS aufzeigen, wobei es in der Literatur viele verschiedene Ansätze gibt, die bestimmte Einsatzgebiete unterschiedlich anordnen.

2.5.1 klassische Betrachtungsweise

In den folgenden Betrachtungen beziehe ich mich auf die Autoren Puppe /Puppe 91/ und Steinmüller/Steinmüller 05/, die eine Einteilung der Expertensystemanwendungsbereiche im Abstand von 14 Jahren vorgenommen haben. Dabei lässt sich erkenn, wie sich bestimmte Vorstellungen in Bereich der XPS durchgesetzt und die Schwerpunkte sich leicht verändert haben.
Puppe unterteilt die Einsatzgebiete von XPS in Interpretation, Diagnostik, Überwachung, Steuerung, Design, Planung und Vorhersage, wobei er die Problemlösungsstrategien in die Diagnose, die Konstruktion und die Simulation einteilt.

Puppe (1988,1991)



Steinmüller (2005)

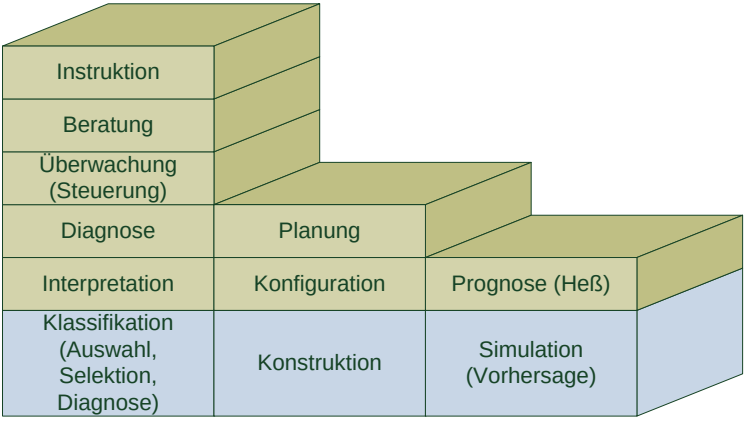


Abbildung 2.8: *Vergleich von Puppe und Steinmüller*

Steinmüller und Puppe unterteilen die XPS beide in drei Problemlösungs-Haupttypen
Klassifikation (Steinmüller) / Diagnostik (Puppe)
Puppe bezeichnet diesen Bereich zwar als Diagnostik geht aber wie Steinmüller davon aus, dass hier aus einer vorgegebenen Menge von Alternativen eine Lösung ausgewählt wird.

Konstruktion (beide)
Während Steinmüller hier nur von einer Entwicklung von Strukturen, die bestimmte Funktionen realisieren, ausgeht ist die Konstruktion für Puppe nur ein Anordnungsproblem. Bestimmte Fakten sind im System an bestimmte Objekte gebunden und werden durch Fragestellungen des XPS so kombiniert, dass die Aufgabenstellung erfüllt wird.
Durch den etwas breiter gefassten Ansatzpunkt kann Steinmüller heute von einer Schnittstelle zu CAE und CAD-Komponenten ausgehen, die zurzeit als Puppe die Einteilung vornahm, noch nicht in der heutigen Form und Leistungsfähigkeit vorlagen. In der Konstruktion fließen natürlich auch bestimmte Berechnungen ein, die überhaupt erst zeigen, ob das System arbeitsfähig ist /Scheer 90//Spur 94/

Simulation (Vorhersage) (beide)
Beide Autoren gehen hier davon aus, dass aus dem Ausgangszustand Folgezustände hergeleitet werden. Ziele - so wird bei Puppe angemerkt - sollten hier nicht erreicht werden, sondern nur die Auswirkungen von Handlungen und Entscheidungen simuliert werden.
Ich empfinde dies als etwas zu kurz gegriffen. Für mich sind Flugsimulationen z. B. eine Mischung zwischen einem Simulationssystem und einer Trainingssystem. Dabei soll das System aber in einen bestimmten Zielzustand überführt werden.
Der Bereich Simulation kann außerdem sehr weit gefasst werden. So kann man auch aus einem Endzustand verschiedene Anfangszustände simulieren oder zeitlich simulieren, wobei man die Veränderung eines Systems betrachtet.
/Heß 02/ folgend würde ich in dieses Feld auch noch die Prognose – Systeme hinzufügen. Bei diesen Systemen versucht man aus gegebener Beobachtung auf die Zukunft zu schließen(z.B. Wetterprognose).

Die angesprochenen Autoren unterteilen diese Obergruppen nun folgendermaßen:

- die Klassifikation/Diagnostik in die Unterbereiche Interpretation, Diagnose, Überwachung (Steuerung), Beratung(Analyse) und Instruktion (Training) und

die Konstruktion in Konfiguration(Design) und Planung.

Die Unterteilung des Bereichs Klassifikation/Diagnostik wird wie folgt näher erläutert:

Interpretation

aus beobachtbaren Daten (z.B. Sensordaten) werden Objekt- oder Situationsbeschreibungen abgeleitet, so wird zum Beispiel aus Probebohren in der Geologie auf Lagerstätten geschlossen.

Diagnose

Aus Beobachtungen (Symptomen) wird auf ein Systemfehler (Krankheiten) geschlossen; zur Beseitigung der Fehler wird eine Therapie (Reparaturvorschlag) generiert.

Nach meiner Meinung müsste man heute auch alle automatisch erfassbaren Daten mit einbeziehen. Eine Steuerung kann in der technischen Diagnostik so eine ganze Reihe von Systemzuständen liefern, die vom Benutzer nicht direkt beobachtbar sind, durch die Steuerung aber verfügbar gemacht werden können.

Überwachung: (Steuerung)

Steinmüller möchte bei einer Abweichung vom vorgegebenen Normalzustand direkt steuernd eingreifen. Puppe unterscheidet hingegen hier zwischen Überwachung, als etwas passivem und der aktiv eingreifenden Steuerung. Mir scheint diese Unterscheidung ebenfalls im Sinne der Mess- und Reglungstechnik korrekt.

Die Überwachung wird dabei als Vergleich von Beobachtungen mit Sollwerten betrachtet. Bei der Überwachung eines Patienten in der Medizin wird das System laufend erhaltene Daten des Patienten mit Sollwerten vergleichen und bei Abweichungen den Arzt informieren.

Puppe erweitert diesen Beobachtungsvorgang bei der Steuerung um das automatische Veranlassen von Aktionen. Dies kann in technischen Systemen etwa bei der Steuerung eines Raumschiffs oder eines Roboters noch hingenommen werden, könnte aber wohl in der Medizin zu juristischen Problemen führen, wenn die Aktion nicht sehr trivial wäre. Als Beispiel nennt /Dabringer 09, S. 19/ hierfür die „automatischen Entwöhnung von Beatmungspatienten in der Intensivmedizin (z.B. Smart Care/PS)“.

Die Bereiche Beratung und Instruktion werden von Puppe nicht betrachtet, haben heute aber große Bedeutung erlangt.

- Beratung: (Analyse)
  Steinmüller fasst hier die Analyse und die Beratung zusammen. Er versteht unter Analyse die Sichtung und Interpretation komplexen und umfangreichen Datenmaterials (z.B. Finanz- und Bankwesen, aber auch im Netz). Als Beratung versteht er „die Auskunftserteilung bzw. Unterstützung bei der Anwendung komplexer Systeme oder Vorschriften“ /Steinmüller 05, S.11/.
  Ich würde hier gerne eine Aufteilung beider Bereiche vorschlagen, denn aus der Analyse folgt nicht automatische eine Beratung. Sie kann auch dazu verwendet werden, etwa Vereinfachungen der Daten durchzuführen, diese zu verändern oder automatisch einzugreifen. Die Beratung braucht auch nicht auf einer Analyse von Daten zu beruhen. Man denke zum Beispiel an Eliza.
- Instruktion: (Training)
  Steinmüller versteht unter diesem Bereich die „Anleitung und Unterweisung auf einem bestimmten Spezialgebiet bzw. in der Bewältigung komplizierter Situationen“/Steinmüller 05, S.11/.
  Auch hier scheint mir eine Aufteilung in einen Bereich Instruktion und einen Bereich Training sinnvoll, denn Instruktion weist einen Menschen an ohne speziell etwas einüben zu müssen. Innerhalb eines Trainings aber können ganz spezielle Dinge eingeübt werden, die auch eine spezielle Peripherie notwendig macht. Heute ist dies mit nicht allzu großem Aufwand auch beim PC möglich.

Die Konstruktion wird so weiter unterteilt:

- Konfigurierung: (Design)
  Beide Autoren verstehen darunter die Zusammenfügen (Konfiguration) von Objekten unter Berücksichtigung besonderer Anforderungen (Voraussetzungen).
  Der Design-Prozess umfasst aber für mich noch mehr als eine pure Verwendung bereits bekannter Objekte zu einem neuen. In einem solchen Prozess müsste man dem Benutzer auch ermöglichen etwas völlig Neues zu erschaffen und dies in die Konfiguration mit einfließen zu lassen. Also zum Beispiel das Entwerfen eines neuen Bauteils. Dazu könnte man zum Beispiel ein CAD-System einbringen und mit dem XPS kommunizieren lassen.

- Planung:
  Die Autoren beschreiben diese beide als „Entwurf einer Folge von Aktionen zum Erreichen eines Zieles“. Denkt man aber auch an Planungen, die nicht zu einem Optimum geführt werden können, etwa in der Tourenplanung oder bei anderen Operations Research Problemen, so kann man hier durchaus auch eine Simulation und Analyse der sich aus ihr ergebenden Situation mit einfließen lassen.

2.5.2 Einteilung nach Zielsetzung und Motivation

Unabhängig von der etwa durch Puppe vorgeschlagenen Einteilung in verschiedene Bereiche sieht Reif /Reif 00/unter Berufung auf Gottlob /Gottlob1990/ Zielsetzung und Motivation von Expertensystemen in:

- der Bereitstellung neuer Dienstleistungen,
- Einsatz von XPS entweder als eigenständiges System oder integriert in Analyse und Diagnosegeräten (etwa in der KFZ-Werkstatt),
- innerhalb der industriellen Produktion soll die Qualität, Sicherheit, Produktivität und Arbeitsbedingungen verbessert werden und
- der Produktionsprozess soll durch Verringerung von Fehleranzahl, Ausschuss und Ressourcenbedarf verbessert werden.

Erstaunlicherweise ist diese Argumentation ausschließlich auf den Einsatz von XPS in der Industrie fokussiert. Das XPS vielleicht auch im Bereich der Forschung zur Überprüfung verschiedener KI-Thesen verwendet werden können, ist 1990 schon fast aus dem Bewusstsein verschwunden. Was hier aber noch nicht vollständig überblickt wurde, sind die enormen Kosten, die die Entwicklung und die Wartung eines XPS macht. Wahrscheinlich müssen wir bei all diesen Motivationen heute noch in Klammern setzen „soweit es sich rechnet“.
Aus den genannten Gründen möchte ich die oben angegebene Aufzählung noch um folgende Anstriche ergänzen und/oder die oben angeführten Zielsetzungen etwas präzisieren:

- Verbesserung der Ausbildung durch die Einbeziehung von XPS.
  Entweder durch die Einbeziehung eines XPS in die Darstellung von Aufbau-, Wirkungsweise und Fehler eines Systems (sowohl in der technischen wie in der medizinischen Diagnose) oder durch die Integration eines XPS in ein Intelligentes Tutorielles System (ITS).

- automatischen Erstellung von technischen Diagnosen und Navigation etwa in der Raumfahrt oder der Robotik,
- Untersuchung und Beherrschung von großen, heterogenen Datenmengen, etwa im Netz.

Zugegebener weise sind die von mir angegebenen Erweiterung auch heute noch eher Forschungsbereiche und rechnen sich wahrscheinlich ebenfalls noch nicht alle.

Inzwischen werden Expertensysteme in den unterschiedlichsten Bereichen eingesetzt. Die Einsatzfelder reichen dabei von der Erdbebenvorhersage (z.B. SPERIL) und Wetterprognose, der Analyse von Informationen von betrieblichen Daten (Datenanalysen), von akustischen (Sprachverarbeitung) und optischen Informationen (Bildverarbeitung), dem Entwurf (Design) eines Bauplans mit Hilfe von CAD-Systemen, Planungssysteme im Vertrieb, in der Beschaffung und Lagerhaltung, in der Produktion und Logistik über die medizinische und technische Diagnose bis zur Risikoüberprüfung bei medizinischen Versicherungen oder die der Kreditvergabe von Banken bis hin zum Einsatz in lehrenden Systemen. /Dabringer 09//Heß 02//Richter 10//Gabriel 08/

2.5.3 Ableitung einer Aufteilung von XPS aus dem Bereich Diagnose

/Wischnewsky 10/ hat für den Bereich der Diagnose eine wie ich finde sehr anschauliche Einteilung von Funktionen für XPS gegeben, die man meiner Meinung nach auch auf andere Bereiche ausdehnen könnte.
Die Systeme erfahren bei Wischnewsky folgende Ausprägung:

- der "Schiedsrichter".
  Unterstützung durch die Nutzung von Referenzmaterial. In der Diagnostik sind dies z.B. Fallsammlungen und Standardfälle. Diese sind auch für alle anderen Bereiche denkbar.
- der "Wachhund".
  Überprüfung von routinemäßig generierten Daten. Wischnewsky nennt hier für die Diagnose z. B. die Patientendokumentation und Labordaten. In allen Bereichen in denen statistische Daten erstellt

werden, etwa bei der Betriebsdatenerfassung, sind solche Systeme einsetzbar.

- der "Erinnerer"
  Unterstützt den Experten im Bereich der weiterführenden Problemlösung, also etwa "Was könnte noch in Frage kommen" oder „was kam den in diesem Zusammenhang schon einmal vor“.

- der "Simulator"
  Hierbei werden Möglichkeiten untersucht, die noch gar nicht eingetreten sind. Dabei können Eingriffsstrategien erarbeitet werden. Auch in der Ausbildung sind solche Systeme sinnvoll. Im Bereich der ITS sind XPS als Komponente eingeplant.
- der "Dolmetscher"
  Als intelligente Schnittstellen zum Beispiel innerhalb einer Dialogschnittstelle zwischen System und Nutzer. Dem Nutzer kann dabei sukzessive eine Fachsprache beigebracht werden.
- der "Lotse"
  Unterstützt den Experten bei seltenen Spezialfällen (z.B. bei der Diagnose bei seltene Komplikationen, ...). und
- der "Konsiliar"
  Entspricht der Konsultation eines Fachmannes aus einem anderen Gebiet. Fast jede betrachtete Domain, die in einem XPS abgebildet sind, besitzen Schnittstellen zu verschieden Fachgebieten. Der Nutzer des XPS kann sich hier mit Wissen über ein fremdes Fachgebiet versorgen, die aber Einfluss auf die betrachtete Domain hat. (siehe hierzu auch /Pfeffer 02/)

/Heß 02/und /Uni Würzburg 04/ erweitert diesen Ansatz meiner Meinung nach noch um die

- Kritikfähige Systeme
  Bei denen ein Expertensystem zur Beurteilung vorgegebener Lösungen befähigt ist.

Konkrete Systeme können dabei natürlich auch Mischformen sein.

Eine weitere Einteilung von XPS wird in /Uni Würzburg 04/ ebenfalls für den Bereich Diagnose vorgenommen. Hier erfolgt eine Einteilung der Systeme nach Initiative & Kommunikationsstil:

• Initiative:
  – System passiv: Benutzer startet die Konsultation und gibt Falldaten ein (ggf. mit Nachfrage durch System)
  – System aktiv: Überwacht einen gegebene Systemzustand und gibt Warnungen oder Vorschlägen aus.

• Kommunikationsstil:
  – Beratungsmodell: Das System schlägt selbst Lösungen vor, die vom Benutzer hinterfragt werden können.
  – Kritikmodell: Das System stellt eine zweite Meinung dar, überprüft und kommentiert Lösungsvorschläge des Benutzers.

2.5.4 Expertensysteme als eine Ausprägung von Informationssystemen

Einen sehr interessante Beschreibung eines XPS nimmt /Bachelier 04/ vor. Für ihn sind Expertensysteme eine Ausprägung von Informationssystemen.
Informationssystem stellen dabei ein Sammelbegriff für Systeme, mit den Hauptfunktionen Aufnahme, Verarbeitung und Wiedergabe von Informationen dar. Eine Unterscheidung zwischen Wissen und Information nimmt Bachelier hier nicht vor. /siehe hierzu Kapitel 3/
Die Informationssysteme (IS) werden dabei in: Datenbank-Systeme, Information-Retrieval-Systeme, natürlich-sprachliche Frage-Antwort-Systeme und wissensbasierte Systeme wie z.B. Experten-Systeme unterschieden.

Formal kann ein IS durch den
Quintupel IS = (A, W, I, V, O)
definiert werden.

Die einzelnen Elemente beschreibt Bachelier wie folgt:

A: Erschließungsfunktion
Methoden zur Aufnahme und Speicherung des Wissens im IS. Typischerweise bei XPS als Wissensakquisition bezeichnet.

W: IS-Inhalt

Das Wissen wird dabei mit Hilfe einer festgelegten Repräsentationssprache abgebildet.

I: Menge aller zugelassenen Inputs

Stellt die Problemformulierungen dar, z.B. die Eingabe von Fehlersymptomen bei einem Diagnose-System. Auch diese müssen natürlich in einer festgelegten Form erfolgen, sonst kann sie das System nicht verarbeiten.

V: Menge aller Verarbeitungsmechanismen

Stellt den Problemlösungs- oder Inferenzprozess des XPS dar

O: Menge aller möglichen Outputs

Ergebnisse des Problemlösungsprozess. Dabei werden zum Beispiel neben einem Text verschiedene andere Medien mit ausgegeben, um den Benutzer zu unterstützen.

2.5.5 Netzwerkbasierte XPS

Unter Einbeziehung von Hypertext und Netzwerkarchitektur führt /Köchert 10/ die Begriffe Finexs ("First kind of network based expert system") und Senexs ("Second kind of network based expert system") ein.
Das Finexs definiert er als ein System, das Wissen auf der Basis von Hypertext oder ähnlichen Repräsentationsformen abbildet. Bei der Problemlösung kommen dabei Suchmaschinen und Robots zum Einsatz.
Das Senexs erweitert den Aufbau des Finexs durch Lernfähigkeit. Finexs und Senexs stellen dabei angemeldete Marken dar. (siehe hierzu auch /Juršič 03/)

3 Von der Daten- zur Wissensverarbeitung

3.1 Daten/Information/Wissen

3.1.1 Begriffsbestimmung

Daten, Information und Wissen stehen - darin sind sich die meisten Autoren einig - im engen Zusammenhang. Ein Bild, das in diesem Zusammenhang immer wieder mit unterschiedlichen Ausprägungen verwendet wird, ist Abbildung ist in einer Pyramidendarstellung (Abbildung 3.1).

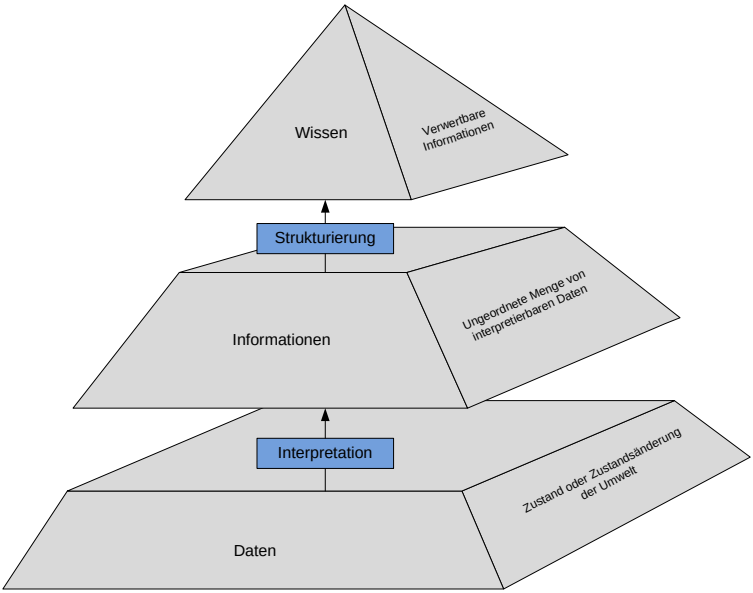


Abbildung 3.1: *Zusammenhang Daten –Information-Wissen /wiki 10 f/*

In der Literatur werden ungezählte verschiedene Definitionen für die im Zusammenhang stehenden Begriffe Daten, Informationen und Wissen gegeben.

Nach /Lehner 09, S. 49/ unterscheidet man in der Literatur (z.B. /Wersig 71/) Definitionen, die davon ausgehen, dass:

- alle Struktur in der Realität als Informationen angenommen werden können;
- man Informationen und Wissen gleichsetzen kann;
- die Informationen als Übertragung von Zeichen betrachtet werden kann;
- die Informationen als Bedeutung von Daten erklären werden kann (Linguistische Definitionen);
- Informationen mit Wirkungen gleichzusetzen sind. Diese Wirkung stellen dabei Ergebnis eines Prozesses, Veränderung von Wissen oder Reduktion von Unsicherheit dar (Empfängerorientierte Definitionen);
- Informationen in Zusammenhang mit Kommunikation oder Handlungen stehen (Prozessorientierte Definitionen).

Ebenfalls auf /Wersig 71/ und /Rauch 82/ bezieht sich /Reif 00/, wenn er die Begriffe anhand einer Definition der Informationswissenschaft erklärt.
In diesem Modell wird davon ausgegangen, dass der Organismus seine Umwelt nicht direkt, sondern mit Hilfe von Sinnesorganen (Perzeptoren) wahrnimmt. Die Perzeptoren geben Reize weiter, durch deren Interpretation der Organismus auf seine Außenwelt schließt und sich ein „internes Außenweltmodell“ aufbaut. Wissen stellt nun die Struktur dieses Modells dar. Denken wird als Operation auf diesem Modell verstanden.
Trifft der Organismus auf einen Weltzustand, für den er über keine Verhaltens-Schemata verfügt (Ungewissheit), so muss er sein internes Modell erst wieder durch neue Aufnahme von weiteren Informationen und Denken anpassen (Reduktion der Ungewissheit). Informationen die hingegen keine Veränderung des Modells notwendig machen werden als redundant betrachtet.

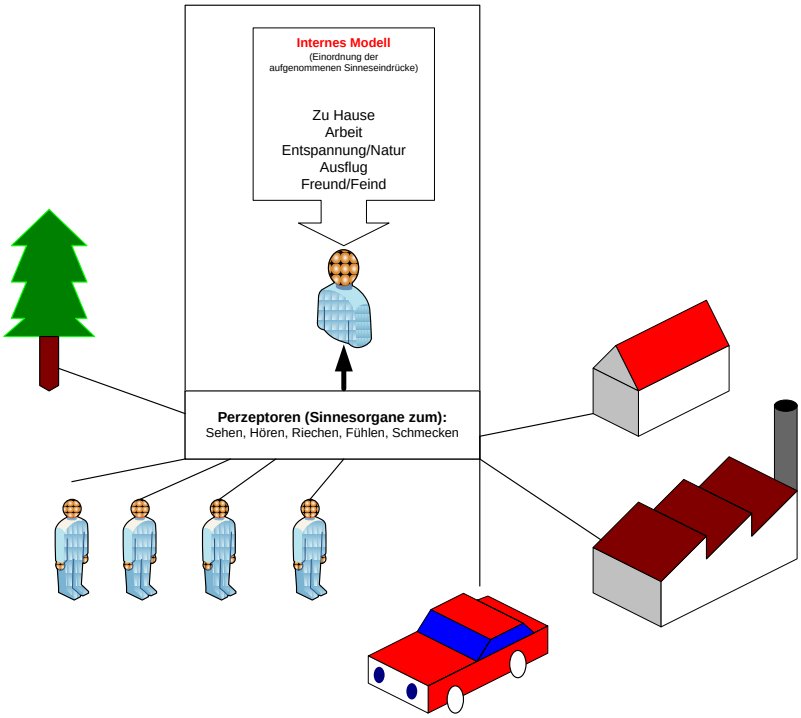


Abbildung 3.2: *Zusammenhang Organismus-Außenwelt-inneres Modell*

Ausgehend von diesem Modell möchte ich nun eine eigene Definition der Begriffe vornehmen.

Definition 3.1: Als Datum wird ein Zustand oder eine Zustandsänderung der Welt verstanden.

Dieser Zustand kann von einem Organismus bemerkbar oder unbemerkbar sein oder er kann erst bemerkt werden, wenn ein bestimmter Schwellwert überschritten wird.
Die Definition ist mit Absicht sehr weit gefasst, weil man glaube ich bei einer Definition von elementaren Begriffen und nicht direkt die rein menschliche Sicht oder die Sicht der Datenverarbeitung einnehmen sollte.
Damit ich ein Datum aufnehmen kann, brauche ich entsprechende Rezeptoren, die das Datum, dann aber bei der Aufnahme interpretieren. Wir sehen genaugenommen nicht das Objekt selbst, sondern die Reflektion des Lichts, dass auf dieses Objekt fällt. Ein Insekt, dass das gleiche Objekt betrachtet würde etwas anderes sehen.
Sobald das Datum von einem Organismus aufgenommen werden kann wird daraus eine Information.

Definition 3.2: Eine Information ist eine Interpretation eines Datums. Sie ist damit vom interpretierenden Organismus und dessen Kontext abhängig.

(sieh hierzu auch /Wiegand 04/)

Dabei kann die Information nicht verwertbar sein, weil zum Beispiel Vorwissen fehlt, redundant sein oder vollständig neu.

Definition 3.3: Als Wissen wird als die Strukturierung von verwertbaren Informationen verstanden.

(siehe unter anderem auch /Wagner 08/)

Damit besitzt der Mensch aber durchaus mehr Informationen als Wissen, weil er auch noch nicht vollständig verstanden Informationen gespeichert hat. Der Mensch hat dann etwas noch nicht vollständig durchdrungen oder noch nicht verstanden, besitzt aber Informationen, die aber noch nicht vollständig zuordenbar sind.
Diese Unterscheidung scheint mir sinnvoll, weil es für bestimmte Informationen einfach noch keinen Anknüpfungspunkt an die Wissensstruktur gibt. Wir sprechen Umgangssprachlich dann auch von Halbwissen.
Aus den Definitionen ergeben sich nun für den Zusammenhang zwischen Informationen und Daten aufgeführten Folgerungen:

- Wir können nur einen Bruchteil der Daten, die vorliegen, aufnehmen. Vielleicht fehlen uns zurzeit einfach nur noch die Rezeptoren. Vor der Erfindung des Mikroskops oder Fernrohrs, waren die

Weltzustände im Mikro- oder Makrokosmos ja schon vorhanden, konnten von uns nur noch nicht aufgenommen werden.

- Daten sind nicht konstant, sie sind hingegen in ständiger Veränderung. Damit kann der Organismus, wenn er Daten aufnimmt nur eine Momentaufnahme erkennen.
- Ob die Daten zu Informationen werden ist vom Kontext des Organismus abhängig, der sie interpretiert. Ein Mensch der schläft nimmt zum Teil andere Interpretationen von Umweltdaten vor, als ein Mensch der wacht. Eine Maus interpretiert Daten anders als ein Mensch usw.
- Das Außenweltmodell ist ein Teil des Wissens. Demgegenüber steht ein Innweltmodell, dass zum Beispiel die emotionale Welt des Menschen darstellt. Ein Wesen wird aber immer von beiden Modellen gesteuert. Das reine Reiz-Reaktions-Schema der Behavioristen ist meiner Meinung nach eindrucksvoll durch die Kognitionspsychologie widerlegt.
- Informationen sind auf die Person bezogen. Verschiedene Personen nehmen Daten auch verschieden auf und interpretieren diese aufgrund ihrer Fähigkeiten.
- Informationen können auch dem eigenen Weltmodell widersprechen.
- Informationen sind zeitabhängig, d. h. sie können veralten.
- Daten besitzen nur Wert wenn sie für den Menschen zugänglich und interpretierbar sind. Wie das Web anschaulich zeigt ist das Vorhandensein von Millionen von Daten für den Menschen noch nicht direkt wertvoll, sie müssen auch von ihm gefunden und interpretiert werden können.
- Wissen ist Macht, nicht Information. Nur wenn die Information in einer verwertbaren Struktur vorliegt kann sie verwendet werden. Man handelt vielleicht mit Informationen und bezahlt auch dafür, nur wenn der „Käufer“ das entsprechende Wissen besitzt kann er auch mit der Information etwas anfangen. Ein Beispiel hierfür wäre das Auflisten der Ergebnisse der Fußball-Bundesliege, ohne Nennung der beiden spielenden Mannschaften.
- Lernen kann als das Verknüpfen von mehreren Informationsbestandteilen verstanden werden. /Wiegand 04/
- Durch die Anwendung der Wissensstruktur können neue Informationsbestandteile, z.B. durch Kombination bekannter

Informationseinheiten entstehen (Wissen als Informationsgenerator). /Wiegand 04/

3.1.2 Austausch von Wissen

Informationen werden aus der Wissensstruktur abgeleitet und wiederum in Daten transformiert, die dann einem anderen Organismus zur Verfügung stehen können, wenn dieser diese Daten auch interpretieren kann.

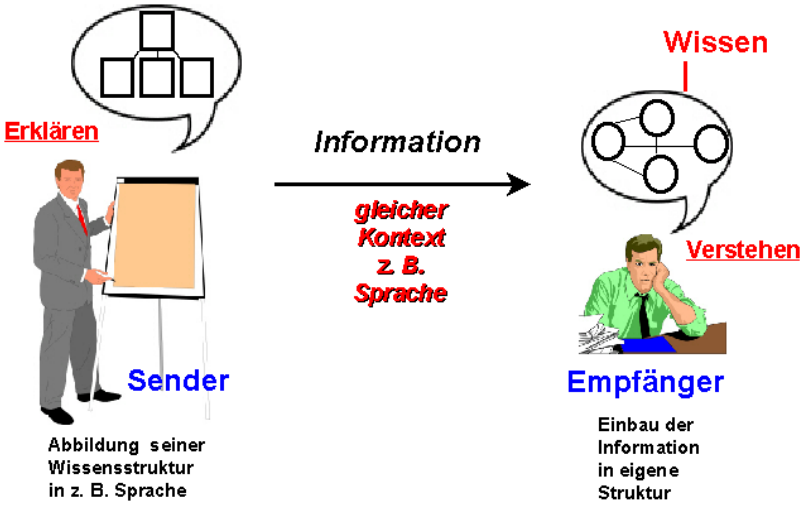


Abbildung 3.3: *Prozess der Wissensvermittlung /Hartmann 01/*

Wie erfolgreich der Informationsaustausch dabei ist hängt dabei von verschiedenen Faktoren ab:

a) dem Sender
Der Sender besitzt Wissen, aus dem er übertragbare Informationen ableiten muss. Die Fähigkeit des Senders aus seiner eigenen Wissensstruktur etwas abzuleiten ist dabei verschieden ausgeprägt (siehe

hierzu auch /Reif 00/). Der Sender kann dabei nicht davon ausgehen, dass seine eigene Wissensstruktur und die des Empfängers der Information gleich sind, daher muss er die Informationen in einer bestimmten Reihenfolge und mit entsprechenden Erklärungen abgehen. Wir können dies als Erklär-Fähigkeit verstehen.

b) dem Kontext

Informationen können nur dann verstanden werden, wenn Sender und Empfänger einen gemeinsamen Kontext besitzen, d. h., dass sie zum Beispiel die gleiche Sprache sprechen und sich über das Thema des Informationsaustauschs einig sind. Beide müssen außerdem willens sein den Informationsaustausch zu vollziehen. Ein Professor kann seinem Studenten nur Wissen vermitteln, wenn dieser nicht gerade eingeschlafen ist oder im Netz "herumservt".

c) Der Empfänger

Der Empfänger muss in der Lage sein die Information auch sinnvoll zu verwerten. Dies ist nicht immer leicht, vor allem weil die Informationen vom Sender sequentialisiert werden müssen, in Wirklichkeit aber vernetzt vorliegen. Dieser Effekt aber ist in der Lehre unumgänglich, weil man dem Lernenden immer nur einen bestimmten Zweig seiner Wissensstruktur darstellen kann. Also etwa „beim ersten Loch kommt der Dampf rein und das zweite Loch krieche ma später“ (Feuerzangenbowle).

In der Literatur unterscheidet man bei der angesprochenen Kommunikation grundsätzlich zwei Schemata (vergleiche hierzu zum Beispiel /Reif 00/).

- Das Sender-Kanal-Empfänger Schemata
  Sender und Empfänger stehen dabei zeitgleich in einer Verbindung. Dabei kann es sich um ein Gespräch, ein Telefonat oder eine Videokonferenz handeln. Die Kommunikation muss aber nicht bidirektional sein (in der Informatik als Vollduplex bezeichnet). Auch eine Radio- oder eine Fernseh-Live-Sendung würde diesem Schemata entsprechen. Da das Gespräch die zahlenmäßig häufigste Form der Kommunikation ist, wird dieses Fall den Großteil aller nicht-technischen Kommunikation ausmachen.

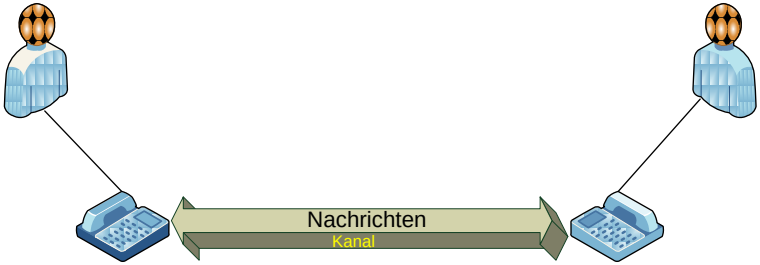


Abbildung 3.4: *Sender-Kanal-Empfänger Schemata*

- Das Sender –Speicher-Schemata

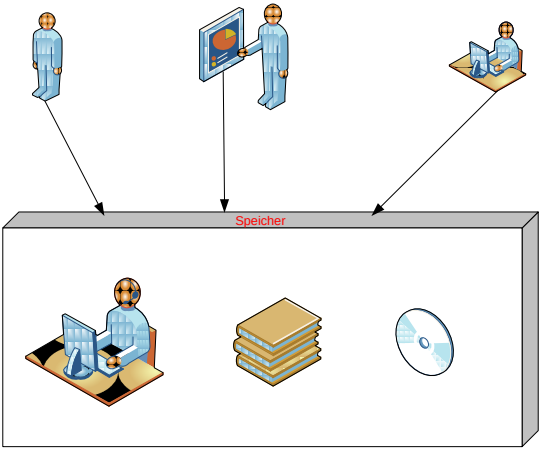


Abbildung 3.5: *Sender-Speicher-Empfänger Schemata*

Innerhalb dieses Schematas werden die Informationen zwischengespeichert und können von einem oder mehreren Empfängern vom Speicher abgerufen werden. Schon ein Buch entspricht einem solchem Schemata, genauso wie ein Film, eine Disk oder ein Computerspeicher und das Web. Der Vorteil ist dabei, dass der Produzent der Information und der Konsument nicht mehr zeitgleich kommunizieren müssen. Der Nachteil ist, dass sich Sender und Empfänger nicht mehr kennen und damit die eigentliche Kommunikation eingeschränkt ist. Stelle ich zum Beispiel während einer Vorlesung fest, dass die Zuhörer mit geöffnetem Mund und einem großen Fragezeichen in den Augen dasitzen, kann ich nochmals erklären, was gemeint ist. Sitzt der Student vor einem Buch, das er nicht versteht, kann er den Autor nicht direkt fragen. Der Produzent einer Information muss sich die Aufbereitung der Information und die Zielgruppe genau überlegen.
Für weitere Betrachtungen von Informationen, Wahrnehmen und Wissen möchte ich auf /Gusik 06/ verweisen.

3.1.3 Der Wert von Information

/Lehner 09/ stellt verschiedene Verfahren dar, nachdem der Wert einer Information ermittelt werden kann.
Als einfachstes nennt er dabei eine subjektive Wertebestimmung, bei der der Nutzer einer Information befragt wird wie viel ihm die Information wert ist. Das Problem bei diesem Vorgehen ist, dass sich der subjektive Wert einer Information im Laufe der Zeit stark ändern kann. Befragt man den Studierenden direkt nach einer Vorlesung über den Wert eines bestimmten Stoffes wird er ganz anders antworten wie der Absolvent und dieser nochmals anders als ein Mensch der mehrere Jahre Berufspraxis hat.
Als objektive Alternative nennt Lehner den Vergleich des Ergebnis oder des Gewinns eines Entscheidungsprozesses mit und ohne die entsprechende Information. Dabei wird die Ergebnis- bzw. Gewinndifferenz als Wert der Information verstanden. Als Problem wird dabei gesehen, dass jeder Entscheidungsprozess einzigartig ist, weil man den situativen Kontext, in dem er durchgeführt wird, in der Realität nicht normieren kann. So gibt es immer Einflüsse, die nicht vorhersehbar und nicht ausschaltbar sind. Die körperliche Leistungsfähigkeit eines Informationsproduzenten oder –Konsumenten variiert zum Beispiel schon an verschiedenen Tagen.

3.2 Wissen

Nach dem *Lateinisches etymologisches Wörterbuch leitet sich der Begriff Wissen* vom althochdeutsch *wizzan* zur indogermanischen Perfektform **woida,* „ich habe gesehen", somit auch „ich weiß" ab. /Walde 38/ /wiki 10 f/
Weitere philosophische Betrachtungswiesen befinden sich in /wiki 10 f/, welches dem interessierten Leser angeraten sei.
Der Zweck von Wissen besteht in der Vorbereitung und Durchführung von Handlungen und Entscheidungen /Wiegand 04/. Die Handlungen und Entscheidungen des Menschen fußen also auf dem von ihm abgebildeten Wissen. Die Effizienz des Menschen bei der Wissensanwendung richtet sich sowohl nach der Quantität wie der Qualität dieser Abbildungsstruktur.
Der Wissensbegriff wird in verschiedenen wissenschaftlichen Disziplinen unterschiedlich gebraucht. So verbindet die Philosophie meist Wissen mit Wahrheit. Demnach kann Wissen nur wahr sein, „falsches" Wissen wird hier als Glaube wahrgenommen. Die Psychologie hält das Wissen für etwas was jemand für wahr hält. Damit kann man aber auch „fehlerhaftes" Wissen erwerben. /Görz 03/
Ab Anfang der siebziger Jahre erkannte man in der KI, dass der Mensch sich bei der Problemlösung eher auf speziell auf den Problembereich zugeschnitte Wissensbestandteile statt auf allgemeine Problemlösungsstrategien bezieht. Dieses „Problembereichsspezifische" Wissen (domain specfic knowledge) bildet seither den Untersuchungsgegenstand im Bereich XPS. /Görz 03/
Ich werde in diese Kapitel verschiedene Klassifikationen von unterschiedlichen Autoren darstellen, die aber immer unter der von Görz eingebrachten Betrachtungsweise gesehen werden sollten.

3.2.1 Philosophischer Ansatz von Ryle u.a.

Ging es im vorherigen Abschnitt um die Definition des Begriff Wissens, so werde ich mich in diesem Kapitel mit den verschiedenen Ansätzen beschäftigen, wie Wissen klassifiziert werden kann, wobei natürlich die spätere Verwendung in wissensbasierten Systemen immer Einfluss auf die Betrachtung besitzt.

Ein sehr interessanter Ansatz befindet sich hier in /Ryle 10/ aus dem Jahre 1969. Er wurde von verschieden anderen Autoren etwa Baumgartner, Dreyfus und Jarz erweitert und teilt Wissen in die Bereiche:

- Faktenwissen
- Anwendungswissen und
- Handlungswissen

ein (Bild 3.6). /Baumgartner 93//Dreyfus 87//Jarz 97/

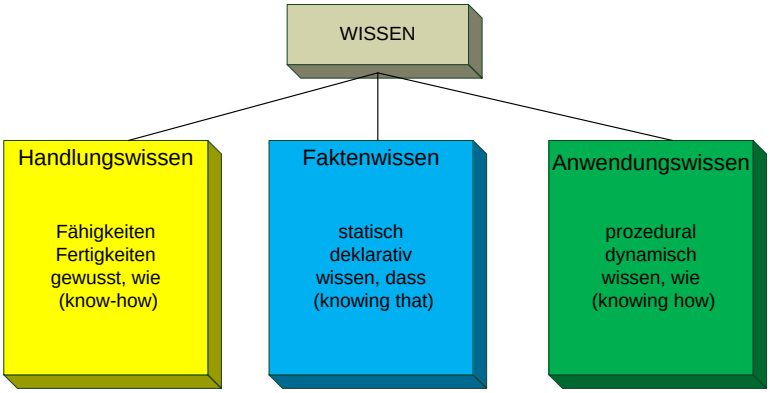


Abbildung 3.6: *Aufteilung des Wissens nach Ryle u.a.*

Faktenwissen gehört dabei neben dem Anwendungswissen zum theoretischen Wissen während Handlungswissen das praktische Wissen, im Sinne von Können, darstellt.

/Jarz 1997, S.72 / definiert Faktenwissen als die Kenntnis von Sachverhalten oder von Aussagen über einen Sachverhalt.

Faktenwissen ist deklarativ und statisch.

„

- deklarativ, weil man damit (vermeintliche) Tatsachen erklärt

- statisch, weil Faktenwissen zwar ergänzt oder erweitert werden kann, aber selbst nicht der Quell neuen Wissens sein kann

„ /Ryle 10/

Das von Ryle als "knowing that" bezeichnet Wissen bezeichnet etwas was so ist oder von dem man annimmt das es so ist. Nach der Meinung der Kognitionspsychologie handelt es sich aber nicht um ein einzelnes Faktum, sonder um einem Netzwerk von sich stützender Fakten (vgl. Jarz 1997, S.73). Typisches Faktenwissen ist z. B. Kenntnis des Periodensystems der Chemie. Wissen, dass sich auf die Kenntnis von Problemlösungen und Prozeduren bezieht wird als Anwendungswissen (vgl./ Jarz 1997, S.73/).
Anwendungswissen wird von Ryle als prozedural und dynamisch verstanden:

„

- prozedural, weil es auf der Kenntnis von Prozeduren zur Problemlösung beruht
- dynamisch, weil als Ergebnis einer Prozedur neues Wissen herauskommen kann (generisches Wissen)

„ /Ryle 10/

Von Ryle wird dieses Wissen als "knowing how" bezeichnet, weil es zur Problemlösung verwendet wird. In diesem Wissen werden die Ursache-Wirkungs-Zusammenhänge erfasst.
Das Handlungswissen versteht sich als „Können“, das aufgrund von Erfahrung gewonnen wurde (daher auch in anderer Literatur als Erfahrungswissen bezeichnet, z.B. auch /Wachsmuth 94/). Dabei lassen geistige Erkenntnisprozesse und (körperliche) Handlungsprozesse voneinander unterscheiden. Da das Handlungswissen auf erworbenen Fertigkeiten beruht, die teilweise nur durch Einüben erlangt werden konnten, ist es schwierig zu erklären (z.B. das Krawattenbinden). Man spricht hier von implizitem Wissen. (vgl. Jarz 1997, S.77 und S.89, siehe hierzu auch /Hubig 04/)
Um handeln zu können, muss man eine Vorstellung vom Problem und seiner Lösung haben, daher stützt es sich auf das Anwendungswissen und dieses kann nur eine Problemlösung finden wenn alle Fakten bekannt sind.

Ich empfehle dieses Modell auch anhand angegebenen URLs zu studieren, in dem zum Beispiel auch auf verschiedene Abbildungsmöglichkeiten etc. eingegangen wird.

3.2.2 Unterteilung nach Gottlob

Dieser Ansatz führt uns jetzt sehr viel näher an den Bereich XPS heran, weil hier schon ein bestimmter Problembereich betrachtet wird. Man erkennt aber auch hier Parallelen zu dem Ansatz von Ryle u.a..
Bezugnehmend auf /Gottlob 90/ teilt /Reif 00/ das in einem WBS verwendete Wissen in vier Kategorien ein:

Fakten zu einem Problembereich

> Diese sind meist in Büchern oder Dokumenten (etwa einer Anlagenbeschreibung, Stücklisten etc.) vorhanden. Sie sind erklärbar und nachvollziehbar. Der Mensch eignet sich z. B. in seiner Ausbildung diese Fakten an. Die oft als Beispiel genannten Naturgesetzte machen aber auch die Problematik klar. Viele der heutigen sicheren Fakten, können sich im Laufe der Zeit als unrichtig erweisen (siehe hierzu auch /Lehner 09/- Domänspezifisches Wissen).

Wissen über die Zusammenhänge der Fakten

> Diese bilden zusammen mit den Fakten den Inhalt eines Wissensstandes über einem Problembereich ab. Auch hier gibt es ein gewisses Restrisiko, dass man Zusammenhänge vielleicht falsch verstanden hat, oder dass bestimmte Fakten innerhalb eines Ursache-Wirkung Zusammenhangs nicht bekannt sind.

Üblicherweise werden Fakten und ihre Zusammenhänge in der Wissensbasis eines XPS dargestellt.

Wissen über Schlußfolgerungsmechanismen und Heuristiken

Dieses Wissen stellt das eigentliche Erfahrungswissen des Experten dar. Da es auf Erfahrung beruht, ist es schwer explizierbar. Innerhalb des KE wird daher versucht gerade dieses Wissen zu extrahieren, was natürlich nie vollständig gelingen kann, weshalb kein noch so gutes XPS den Experten vollständig ersetzen kann. Da sich Erfahrungen nicht nur auf den Kernbereich eines Problems beschränken lassen, findet man hier auch unvollständige Informationen oder solche, die in andere Bereiche verweisen.

Strategisches Wissen

Stellt Problemlösungswissen dar, also wie man effizient Probleme löst. Auch dieses wird im Laufe der Jahre durch Problemlösung im Bereich vom Experten angesammelt.

Wissen über Schlussfolgerungen und Heuristiken und Strategisches Wissen müssen:

- teilweise durch die Struktur der Wissensbasis selbst abgebildet werden (etwa von Knowledge Engineer festgelegte Lösungsschritte und die Reihenfolge der untersuchten Wissensbasis-Teile) oder
- werden in der Problemlösungskomponente als Meta-Wissen abgebildet. (siehe hierzu Kapitel 4)

3.2.3 Erweiterung des Wissens-Begriffs nach Lehner

/Lehner 09/ hat den oben beschriebenen Wissensbegriff noch durch andere wie mir scheint wichtige Begriffe erweitert. So kennt er auch folgende Wissensarten:

soziales Wissen
welches die soziale Fertigkeiten und Kompetenzen enthält, also das Wissen über das menschliche Miteinander. Dabei wird zwischen interpersoneller Kompetenz (Fremdwahrnehmung) und intrapersoneller Kompetenz (Selbstwahrnehmung) unterschieden.
Metakognitives Wissen
Bezeichnet Wissen, dass die Lern- und Denkvorgängen kontrolliert und steuert. Es bezieht sich nicht auf ein spezielles Problem, sondern stellt Strategien zur Verfügung, die bei der Problemlösung aktiviert werden.

Strategisches Wissen (hier im Gegensatz zu Reif und Gottlob)
Ist Wissen für Probleme, die man mit dem allgemeinen Metakognitiven Wissen nicht erfassen kann (z. B. Heuristiken und spezielle Problemlösungsstrategien).

Wissen kann aber auch auf ganze Gruppen von Menschen bezogen werden (siehe hierzu auch /Wachsmuth 94/). In diesem Zusammenhang nennt Lehner:

Organisatorisches Wissen
Das ist Wissen, dass über eine Organisation verteilt ist. Je nach Entscheidungsverantwortung müssen die verschiedenen Entscheidungsträger dabei auf dieses Wissen zugreifen können. In der Organisation muss außerdem einen Konsens über dieses Wissen bestehen, d. h. der Inhalt des Wissens muss von allen Organisationsmitgliedern akzeptiert werden. Dies erfolgt innerhalb der Organisation über einen Kommunikations- und Abstimmungsprozess. Damit es in einer Organisation optimal wirken kann, muss es in sich stimmig und widerspruchsfrei sein.
Kollektives Wissen
Stellt personenunabhängiges Wissen dar, auf das prinzipiell alle Mitglieder einer Gemeinschaft zugreifen können. Es steht in Form von Regelwerken für soziale oder organisatorische Systeme zur Verfügung (z. B. Leitlinien, Standards, Proessbeschreibungen, Routinen, Rezepturen, Traditionen, Sozialverhalten). Im betrieblichen Bereich nennt Lehner hier auch noch Informationen über Kunden, Lieferanten, Mitbewerber und Trends. /Bimazubute 05/ nennt zum Beispiel hier die Leistungen einer Mannschaft in einem Teamsport, die über Summe der Einzelleistungen hinausgeht. Er beschreibt auch sehr anschaulich wie sich dieses Wissen bilden kann. /Bimazubute 05/

/Abecker 00/ hat in diesem Zusammenhang das Wissen auf verschiedenen Ebenen definiert. Er unterscheidet dabei die individuelle Ebene(Intuition, Fähigkeiten, Kenntnisse und Erwartungen), die Gruppen-Ebene (Routinen, Rollenverteilung, geteilte Sprache und komplementäre Fähigkeiten) und die organisatorische Ebene (Kernkompetenzen, Mythen, geheime Spielregeln, Verträge, elektronische Wissensbasis, und externe Vernetzung), wobei bei diesen Unterteilungen offensichtlich nur Beispiele angeben und diese beliebig erweiterbar sind.

In der Literatur finden sich auch noch andere Aufteilungen des Wissens, die aber nur bestimmte Bereiche nochmals detaillieren, ohne ganz neue Erkenntnisse zu bringen so differenziert z. B. /Budin 04/ nochmals in Handlungswissen, Orientierungswissen, Prozesswissen, Problemwissen, Sprachwissen, Kommunikationswissen, Sachwissen, Referenzwissen, Kulturelles Wissen, Kollektives vs. individuelles Wissen, Objektives vs. subjektives Wissen, Hypothetisches vs. sicheres Wissen, Alltagswissen, Welt- und Allgemeinwissen, Fach- und Expertenwissen.

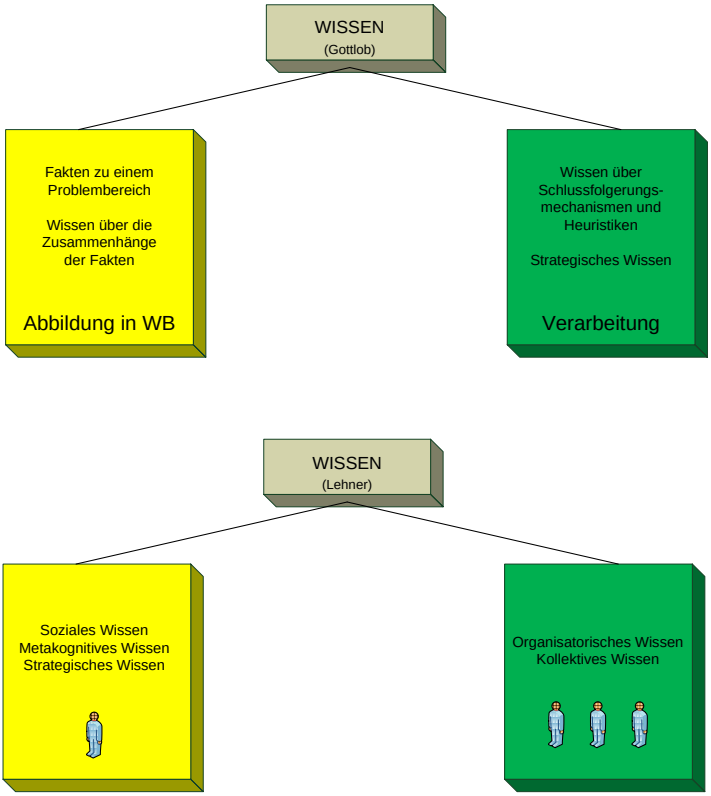


Abbildung 3.7: *weitere Unterteilung des Wissens*

3.2.4 Die Rolle der Erfahrung bei der Anwendung von Wissen

/Wachsmuth 94/ weist darauf hin, dass zwischen dem in Büchern oder anderen Medien festgehaltenen Wissen und dem bei der konkreten Problemlösung eines Experten verwendeten Wissens kein einfacher Zusammenhang besteht. So beeinflusst das durch den Expertenaufgenommenen Fachwissen zwar die Art der Problemlösung, gleicher Erfolge können sich aber auch bei unterschiedlichen Experten mit unterschiedlich angelerntem Wissen ergeben. Die Qualität eines Experten wird maßgeblich durch sein Erfahrungswissen in diesem Gebiet geprägt, dass über das Kernwissen des Fachs, wie es in Büchern abgebildet ist, hinausgeht und vom Experten in einer eigenen Wissensstruktur verfügbar gehalten wird. Durch diese eigene Struktur wird das Wissen im Bedarfsfall schneller zugreifbar und lässt sich genauer auf einen Problemfall anwenden. Dies ist mit ein Grund warum der Experten einem guten Anfänger, welcher nur über angelerntes Wissen verfügt, in Bezug auf Qualität, Exaktheit und Detaillierungsgrad meist überlegen ist.
Erfahrungswissen wird durch das Erleben vieler verschiedener Situationen mit verschiedenen Kontexten in einem Problembereich erlangt. Dabei kann der Experte eine Situation eher ganzheitlich erfassen. Er bekommt einen geübten Blick für den gesamten Kontext (in der Medizin der sogenannte „klinische Blick"). Ein geübter Schachspieler erfasst zum Beispiel das Muster einer Stellung auf dem Schachbrett und versucht nicht jede einzelne Figur zu analysieren. Das Lehrbuchwissen wird vom Experten aufgrund dieser Erfahrungen umorganisiert und gewertet. Wer sich zum Beispiel jemals als praktisch Lehrender mit Erziehungswissenschaften beschäftigt hat, wird wissen, dass sich nicht jedes theoretische Modell einer Situation in die Praxis überführen lässt. Der Experte erweiterte das Lehrbuchwissen um eigene praktische Anwendungen im Kontext.
Neben der durch die Erfahrung angepassten Wissensstruktur des Wissens (von Wachsmuth als Kognitive Gliederung bezeichnet) zeichnet sich der Experte auch durch eine eingenommene Perspektive bei der Aufgabenbewältigung aus. Da der Experte die einzelnen Informationen einer Situation entsprechend seiner Erfahrung wertet, kommt es zu einer gerichteten Wahrnehmung (zum Beispiel der Schachspieler, der nicht mehr jede Figur einzeln bewertet, sondert ihren Zusammenhang) und allein dadurch wird schon ein Teil der Problemlösung erreicht.

Experten handeln schnell und flüssig, ohne dabei allerdings bewusst auf ihr Wissen zuzugreifen. Das „Warum“ ist dabei nicht immer vom Experten zu erklären. Innerhalb des Knowledge Engineering führt dies immer wieder zu Problemen. Zum Teil muss der Knowledge Engineer versuchendem dem Experten sein Handeln erst bewusst zu machen. Die Erfassung solchen Handlungswissens macht häufig die Begleitung und Beobachtung des Experten notwendig, weil es theoretisch nicht ableitbar ist. (siehe hierzu auch /Computerwoche 87/)
Da Experten auch einer Gesellschaft angehören, besitzen sie auch das von dieser Gesellschaft als allgemein gültig betrachtete Wissen (z.B. Wissen über Raum und Zeit, einfache physikalische und chemische Effekte, Sozialverhalten …). Dieses Wissens wird aber ebenfalls zur Problemlösung herangezogen. 1994 sieht Wagsmuth die Untersuchung des Allgemeinwissens innerhalb der XPS-Technologie noch am Anfang stehen. Während meiner Recherche zu diesem Buch bin ich hier ebenfalls auf keine nennenswerte Erfolge gestoßen. Ein Problem bei der Einbeziehung von Allgemeinwissen mag schon allein die Menge und Variation dieses Wissens sein, dass selbst bei Menschen der gleichen Gemeinschaft variieren kann. Die Erfolge bei Expertensystemen stellten sich vor allem dann ein, wenn man sich auf ein Spezialgebiet beschränkte.

3.2.5 Wissensrepräsentation

In einem WBS wird Wissen, das in Zusammenhang zu einem abzubildenden Problembereich steht abgebildet und in eine einheitliche Form gebracht. Diese Wissensrepräsentation ermöglicht es dem System das Wissen zu verarbeiten. Das Wissen wird im System in einer Wissensbasis zusammengefasst. Das in der Wissensbasis abgebildete Wissen kann nicht mehr nur einem einzelnen Bereich der oben angesprochenen Klassifikationen zugeordnet werden, sondern umfasst Elemente aus allen angesprochenen Klassen, so sie für die entsprechende Problemlösung wichtig sind. /Wagner 08/
Die Erstellung einer solchen Wissensbasis erfolgt im Rahmen des Knowledge Engineerings durch ein Team von Knowledge Engineers (siehe Kapitel 6). Das Knowledge Engineering kann dabei grundsätzlich in die Schritte Wissensakquisition, Wissensmodellierung und Wissensoperationalisierung unterteilt werden. /Wagner 08/

Grundsätzlich lassen sich die drei Hauptfunktionalitäten *Wissensrepräsentation*, *Wissensverarbeitung* und *Wissensableitung bei einem WBS unterscheiden.* /Wagner 08/
In der KI fasst man unter Wissensrepräsentationsform vor allem Regeln, Fälle, Frames, semantische Netze, constraints, neuronale Netze und daraus abgeleitete hybride Formen. Hierbei sei darauf hingewiesen, dass diese Aufzählung nur die grundsätzlichen Repräsentationsformen darstellen. Diese müssen um überhaupt verarbeitet werden zu können in eine eigens auf den Problemfall bezogenen Wissensrepräsentations-Sprache (WRS) abgebildet werden. Zur Einführung in die verschiedenen Wissensrepräsentationsformen möchte ich /Hartmann 09/ empfehlen.
In diesem Buch verwende ich zur Abbildung des Wissens beispielhaft eine hybride Wissensrepräsentation mit einer Wissensrepräsentations-Sprache (WRS), welche in Zusammenhang mit meiner Vorlesung zu WBS innerhalb eines Praktikums entwickelt wurde. Mir kommt es hier nicht auf die Aufzählung aller möglichen Repräsentations-Formen oder WRS an, sondern um die Darstellung wie diese verwendet werden können. (siehe Kapitel 5 und Anhang B).

3.3 Wissens-Management

Dieses Kapitel führt eigentlich über das Thema des Buches hinaus. Ich möchte dennoch auch diesen Bereich der Wissensverarbeitung kurz ansprechen, weil er die Erkenntnisse der KI mit anderen Bereichen etwa der BWL verbindet und auch monetär interessant werden lässt. Schon aus den etwas klobigen Definitionen lässt sich ermitteln, dass Industrie und Wirtschaft mit diesem Begriff große Potentiale verbinden. Hier kann natürlich nur ein ganz kurzer Abriss gegeben werden, daher möchte ich den interessierten Leser auch auf die angegebene Literatur verweisen.

3.3.1 Definition und Anwendungsbereiche

„Wissensmanagement“ unterstütz den Entscheidungsträger bei dem Erwerb, Zugriff, Nutzung und Weitergabe von Wissen. Dabei werden z. B. Methoden des Knowledge Engineerings verwendet, um die Verwaltung, den Zugriff und die explizite Verarbeitung des Wissens zu gestalten. /Wiegand 04/.
Die folgenden Definitionen sind /Lehner 09/ entnommen, der hierbei auf weitere Literatur Bezug nimmt.

Er führt folgende sich ergänzende Definitionen an:

„Das WM umfasst das Management der Daten-, Informations- und Wissensverarbeitung im Unternehmen…Wissen und Informationen werden dabei als grundsätzlich handhabbare Objekte angesehen, die direkt oder indirekt über Wissens- und Informationsträger in materieller (Daten-)Form vorliegen. WM beschränkt sich dabei aber nicht nur auf den technischen Problemkreis, wie das traditionelle Daten- und Informationsmanagements, sondern es verwaltet insbesondere auch die personellen und institutionellen Wissenspotentiale und deren Verarbeitung.“ /Kleinhans 1989, S. 26/

Kleinhans nimmt meiner Meinung nach damit einfach nur eine Erweiterung der herkömmlichen Datenverarbeitung vor und integriert die Wissensverarbeitung, weil sich auch die Datenverarbeitung meiner Meinung nach nicht ausschliesslich auch technische Probleme bezieht.

„Organisationales WM meint die Gesamtheit korporativer Strategien zur Schaffung einer „intelligenten" Organisation. Mit Blick auf Personen geht es um das organisationweite Niveau der Kompetenz, Ausbildung und Lernfähigkeit der Mitglieder; bezüglich der Organisation als System steht die Schaffung. Nutzung und Entwicklung der kollektiven Intelligenz und des „collective mind" in Frage; und hinsichtlich der technologischen Infrastruktur geht es vor allem darum, ob und wie effizient die Organisation eine zu ihrer Operationsweise kongeniale Kommunikations- und Informationsstruktur nutzt" /Willke 1996, S. 280/

Willke begrenzt hingegen das WM auf die Organisation, während Reinmann-Rothmeier und Mandl das WM auf die gesamte menschliche Welt erweitern, in dem Sie WM beschreiben als:

- Informationen verbreiten, selektieren und bewerten,
- Informationen in einem Kontext einbetten und mit Bedeutungen versehen,
- Aus Informationen Wissens konstruieren und neues Wissen entwickeln,
- Wissensinhalte miteinander verknüpfen und Wissensnetze bilden,
- Wissens bewahren, strukturieren und aktualisieren,
- Wissen weitergeben, vermitteln und verteilen,
- Wissen austauschen und gegenseitig ergänzen,
- Wissen anwendenden und umsetzen und
- Wissensbasiertes Handeln bewerten und daraus neues Wissen entwickeln.

/Reinmann-Rothmeier 1997, S. 22/

Vielleicht die kürzeste und treffendste Definition ist allerdings:
WM ist „der gesamte Prozess von der Wissenserfassung, Änderung bis hin zum Finden und Strukturieren von Wissen" /Christmann-Jacoby 1997, S. 23/

/Lehner 09/ sieht mit Verweis auf weiterführende Literatur prinzipiell die Einsatzgebiete des WM in:

- „der Zielgerichtet und geplante Wissensversorgung einer Organisation;
- Umgang mit Ressource Wissen als knappes Gut
- Management der Kosten und Leistungspotenziale von Wissen;
- Management der Wissensquellen;

- Unterstützung (technische und nicht-technische) Systeme der Wissensproduktion, -reproduktion, - distribution, -verwertung und des Wissensflusses“

/Lehner 09, S.xy/

WM hat dabei als verschiedene, sich ergänzende und durchdringende Untersuchungsobjekte:

- Gesellschaft,
- Städte, Region, Länder,
- Organisation, Institutionen und Unternehmen und
- Personen bzw. Individuen. /Lehner 09/

Im Untersuchungsbereich *Gesellschaft* können die Wissensströme zwischen den verschiedenen gesellschaftlichen Gruppen und Institutionen betrachtet werden. Dabei wird man wahrscheinlich sehr viele verschiedene Wissenschaftsbereiche etwa die Sozial- und Kulturwissenschaften mit einbeziehen müssen. Sie nur als übergeordneten Rahmen der anderen Bereiche zu betrachten, wie das Lehner tut, scheint mir zu kurz gegriffen.
Im Bereich *Städte, Region, Länder* werden aus betriebswirtschaftlicher Sicht ganze Wirtschaftsräume betrachtet. Man hat hier z. B. die Begriffe „Knowledge Cietes“, Knowledge Regions“ oder „Knowledge Countries“ eingeführt (Beispiele dafür sind München, Barcelona …). Man möchte hier besonders wettbewerbsfähige Gebiete schaffen, in dem man ganz bewusst die „Ansiedlung und Vernetzung wissensintensiver Branchen inklusive des dafür notwendigen Aufbaus finanzieller, rechtlicher, infrastruktureller etc. Rahmenbedingungen“ /Lehner 09, S. xy/ schafft.
Nimmt man auch hier die etwas breiter angelegten Definitionen von WM, so könnte man durchaus auch eine soziale Betrachtungsweise in den Gebieten vornehmen, um so dass Zusammenleben der Menschen zu verbessern. Auch dabei spielt Wissen der verschiedenen Gruppen eine entscheidende Rolle.
Der erste Untersuchungskontext von WM war der Bereich *Organisation, Institutionen und Unternehmen.* Heute wird er aber auch auf andere Bereiche erweitert. Lehner nennt hier als Beispiel Haushalte, Behörden, Bildungseinrichtungen und Non-Profit-Organisationen./Dahlmann 06/
Der Mensch sollte natürlich bei jeder Untersuchung im Mittelpunkt stehen. Er ist es der Wissen aufbaut und aufgrund dieses Wissens auch handelt. Im Untersuchungsgegenstand *Person- Individuum* schließt sich meiner Meinung nach wieder der Kreis hin zur KI. Denn ein wesentlicher Anteil der

Forschungsinteressen der KI besteht ja darin, wie der Mensch sein Wissen verwaltet und anwendet.

3.3.2 Rollen im Wissensmanagement

/Bimazubute 05/ beschreibt verschiedene Funktionen innerhalb eines WM-Prozesses:

Wissensmanager (Wissensdirektor, Chief Knowledge Office)

Dieser hat die Aufgabe das WM zur Realisierung von Unternehmenszielen einzusetzen und die geplanten Maßnahmen zu überwachen. Planabweichungen werden von ihm analysiert und gegebenenfalls wird das WM angepasst.

Wissenslieferanten (Fachexperten)

Diese stellen der entsprechenden Gemeinschaft oder Organisation ihre Erfahrungen in einer bestimmten Domäne zur Verfügung. Damit ergeben sich aber alle Probleme, die man im Bereich des Knowlegde Engineerings ebenfalls hat. Unter anderen die Bedeutung der Experten für die Gemeinschaft und die damit verbundene notorische Zeitnot, oder die Schwierigkeit Erfahrungswissen zu explizieren. Dem Fachexperten muss vor allem klargemacht werden welche Vorteile das WM hat, sonst ist er eher geneigt den Aufwand zu meiden.

Wissensnutzer

Stellen die eigentlichen Nutznießer des WM dar. Über das verfügbare Wissen, sind sie in der Lage ihren Aufgaben besser nachzukommen. Mit dem WM gemachte Erfahrungen können dabei genutzt werden, um eine Verbesserung des Systems zu erreichen. Wissensnutzer aber sind keine Experten, dazu fehlt ihnen einfach die Erfahrung. Diese Rolle sollte meiner Meinung nach für den Wissenserwerb auch nicht überschätzt werden. Ohne fortwährende Arbeit mit Fachexperten wird man keine zufriedenstellenden Systeme realisieren können.

Wissensmediator

Er ist für die Qualität des zur Verfügung gestellten Wissens verantwortlich und versucht Lücken und Inkonsistenzen in der Wissensbasis zu finden. Da er nach Bimazubute auch für die Strukturierung und den Aufbau der Wissensbasis verantwortlich zeigt, kann er mit dem Knowledge Engineer im Bereich der WBS verglichen werden. Die verwendeten Wissensbasen haben aber, auch darauf möchte ich hinweisen, vor allem wenn sie größer werden, die gleichen Probleme wie im Bereich der WBS. Ihre Korrektheit und Vollständigkeit ist schwer nachzuweisen und sie besitzen den Hang zur Überalterung, wenn sie nicht ständig gewartet werden.

3.3.3 technologischer und humanorientierter Ansatz von WM

In der /Gronau 08/ werden der technologische und humanorientierte Ansatz von WM voneinander unterschieden. Im Fokus des technologischen Ansatzes steht dabei die informationstechnische Abbildung des Wissens, z. B. in einem Computer, bei der humanorientierten Betrachtungsweise steht der Wissensträger Mensch im Mittelpunkt.
Als weitere Unterteilung wird eine systemorientierte und eine prozessorientierte Betrachtungsweise vorgenommen. Das systemorientierte WM versucht dabei Wissen zu ermitteln, es sinnvoll zu repräsentieren und in einem entsprechenden Informationssystem zur Verfügung zu stellen. Die prozessorientierte Herangehensweise richtet sich eher an Prozessverläufen aus (z. B. einem Geschäftsprozess) /Remus 02/.
Bei der Abbildung des Wissens treffen wir auf alle bereits oben angesprochenen Wissensarten und der dort dargestellten Problematik (vergleiche hierzu implizites - hier als stillschweigendes Wissen bezeichnet - und explizites Wissen)
Als Beispiel einer Modellierungssprache wird in der Quelle auf die *Knowledge Modelling Description Language* (KMDL) /Gronau 05/ verwiesen, die vor allem in der Abbildung von Wissensflüssen spezialisiert ist und auch stillschweigendes Wissen berücksichtigen kann.

4 Aufbau eines Expertensystems

In diesem Kapitel werde ich auf die generelle Struktur eines Expertensystems und seiner Komponenten eingehen. Kapitel 5 zeigt dann einen speziellen Aufbau eines Expertensystems, den ich in meinen Vorlesungen verwende.

4.1 Überblick und Einordnung

4.1.1 Einteilung der Systeme aufgrund der Interaktion Mensch-Umwelt

Die Beziehung zwischen Mensch und Umwelt und die daraus resultierenden Klassen von Expertensystemen wird in der folgenden Abbildung dargestellt. Ich habe dabei dasWort Beratung, das bei Reif verwendet wird, durch Konsultation ersetzt, um einen deutlichen Unterschiede zu der in Kapitel 2.5.1 verwendeten Einteilung von Expertensystemen aufzuzeigen. /Reif00/

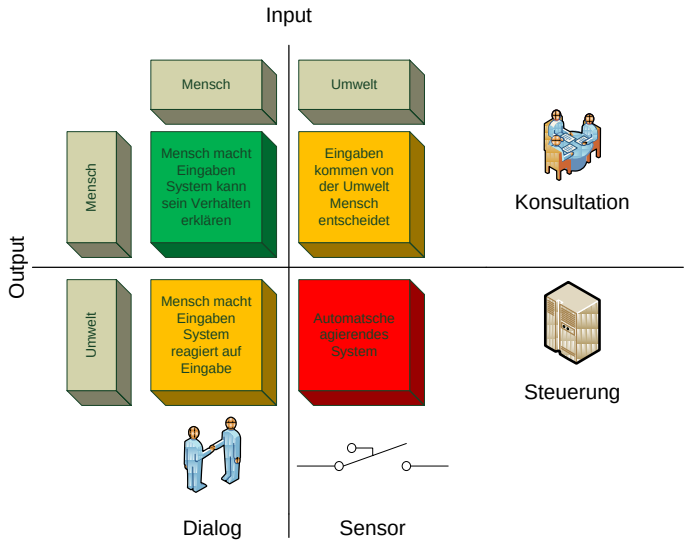


Abbildung 4.1: *XPS-Einteilung aufgrund der Interaktion Mensch/Umwelt (/Reif 00/, /Gottlob 90/)*

Nach Reif und Gottlob kann man im Bereich der XPS bei der Interaktion des Menschen mit der Umwelt zwischen vier verschiedenen Arten von Systemen unterscheiden:

- dem Dialoggeführten Konsultationssystem, in dem der Mensch die Daten jedenfalls zum Teil eingibt und die Problemlösung einem anderen Menschen angezeigt wird, der diese dann überprüfen kann;
- dem Dialoggeführten Steuerungssystem, in dem der Mensch über das Expertensystem direkt auf die Umwelt einwirkt, wobei die Auswirkungen der Entscheidungen innerhalb des Dialogs angezeigt werden können;
- dem Sensorgeführten Konsultationssystem, in dem die Daten automatisch aufgenommen werden, der Mensch aber letztendlich die Entscheidung über die Verwendung der Daten hat und diese entsprechend seiner eigenen Erfahrungen auch überprüfen oder sich erklären lassen kann und
- dem Sensorgeführte Steuerungssystem, bei dem der Mensch nicht mehr einbezogen ist. Diese automatischen Systeme sollten natürlich nur in unkritischen Bereichen eingesetzt werden. Was passieren kann wenn man Systemen Entscheidungen überlässt konnte man beim automatischen Kauf von Aktien durch entsprechende Systeme sehen, bei der Steuerung eines Kernkraftwerks hätte dies weitreichendere Folgen. /Reif 00/

Die meisten XPS zählen heute zur Klasse der Dialoggeführten Konsultationssysteme.

XPS sollen laut /Richter 10/ das Verhalten eines menschlichen Experten simulieren, wobei das Expertenwissen und die Problemlösungsstrategien getrennt verwaltet werden sollen. Das System soll dabei erklärungsfähig, flexibel, benutzerfreundlich und kompetent sein, in Bereichen eingesetzt werden können, in denen die Verarbeitung auf diffusem, unsicherem oder unvollständigem Wissen beruht und heuristische Problemlösungen einbeziehen.

/Gabriel 08/ beschreibt XPS als interaktive computergestützte Problemlösungssysteme, die auf abgegrenzten Teilbereichen des menschlichen Wissens eingesetzt werden. Die Systeme leiten die Problemlösung über Schlussfolgerungen aus gespeichertem Wissen ab, führen einen „natürlich sprachlichen Dialog“, erklären ihr Verhalten und sind lernfähig. Die Meinung von Gabriel, dass sie dabei „intelligentes“ Verhalten zeigen, muss man im Sinne der Diskussion innerhalb der Künstlichen Intelligenz verstehen (siehe hierzu auch Kapitel 2.1.4)

/Reif 00/ sieht in der Verbesserung der Qualität des Produktionsprozesses und der einen Hauptmotivationsgrund zur Verwendung von XPS. Dazu müssen diese aber in den Produktionsprozess eingepasst sein (z. B. durch die unten beschriebenen Zusatzkomponenten).
Um alle Anforderungen auch nur annähernd erfüllen zu können, muss ein solches System einen bestimmten Aufbau und gegeneinander abgrenzbare Komponenten besitzen. Vor der Beschreibung des konkreten Aufbaus eines solchen Systems möchte aber ich zunächst auf einige Probleme hinweisen, wie sie aus dem Einsatz der Sdysteme in der Praxis ergeben.

4.1.2 Problemfelder

Bei der Realisierung von XPS können verschiedene Problemfelder ausgemacht werden. Neben der Erfahrung, die das Entwicklungsteam mitbringt, beeinflusst sowohl die Auswahl möglicher Entwicklungswerkzeuge, wie die Bereitstellung der Experten die Qualität des Systems erheblich.
Von der Meinung des Jahres 1987, dass XPS vor allem Laien bei der Bewältigung von Problemen helfen sollten ist man heute abgekommen /Computerwoche 87/. Allein die verwendete Fachsprache kann von vielen Laien überhaupt nicht verstanden. XPS stellen heute eher eine Hilfe für Menschen mit einem Grundwissen dar.
Wo die Erfahrung von Experten abgebildet werden kann und es keine genauen Algorithmen oder nur solche, die zu unvertretbarem Rechenaufwand führen zur Problemlösung gibt, lohnt sich der wirtschaftliche Einsatz der XPS-Technologie /Computerwoche 87/.
Neben den Schwierigkeiten überhaupt Experten zu finden, die ihr Vorgehen bei der Lösung eines Problems adäquat beschreiben können, soll das System sowohl den Großteil aller anfallenden Probleme in im Einsatzfeld lösen, wie darüber hinaus auch so viel Qualität aufweisen, dass der Anwender zufrieden ist (siehe hier auch Kapitel 6). /Cawsey 03/
Die Entwicklung eines XPS muss sich darüber hinaus auch finanziell lohnen. Wenn man bedenkt, dass die Entwicklung eines XPS schnell mehrere 100 T€ kosten kann, liegt ein enormer Druck auf den Entwicklern und den Managern, welche die Einführung des Systems befördern. /Cawsey 03/ nennt in diesem Zusammenhang eine Reihe von Problemen, die man bei der Entwicklung von XPS berücksichtigen sollte:

- Eine realistische Einschätzung der Kosten und Nutzen.

Gerade in einem technischen anspruchsvollen Bereich „toben“ sich die Wissenschaftler gerne aus, ohne dass die Kosten auch dem Nutzen des Betreibers entsprechen. Einfache Entscheidungshilfen können aber auch mit entsprechenden Darstellungen (etwa Flussdiagrammen) sinnvoll abgebildet werden;

- Verfügbarkeit der Spezialisten zum Wissensgebiet.

  Während der Entwicklung muss immer wieder auf die Experten vor Ort und eventuell auch auf Kunden und Zulieferer zugegriffen werden. Dies belastet die entsprechende Firma auch im Arbeitsablauf;

- Die Auswahl der Domäne.

  So gab es zwar schon Versuche auch physische Fertigkeiten zu unterstützen, der Einsatz scheint aber durch noch eingeschränkten Sensoren und Aktoren fraglich;

- Menge des zu beachtenden Wissens.

  XPS sind dort besonders effektiv, wo fachlich eingeschränktes Spezialwissen abgebildet werden soll. Wo sehr viel auf Allgemeinwissen zurückgegriffen wird, müssen große Datenmengen, die zu dem noch unstrukturiert vorliegen können, abgebildet werden. Die Meinung, dass sich vor allem Problemlösungen eignen, bei welcher der Experte nur eine beschränkte Zeit (Casew nennt hier eine Stunde) zur Problemlösung verwenden muss, teile ich hier nicht. Der Problemlösungsvorgang muss dem Anwender nur deutlich gemacht werden und in Unterschritte zerteilbar sein. Die oft beobachtete Ungeduld beim Einsatz von Computern in allen Einsatzgebieten kann nicht der XPS-Technologie angelastet werden.

/Schneider 10/ weißt darüber hinaus darauf hin, dass XPS nicht in allen Fällen korrekte Problemlösungen liefern. Vor allem bei großen Wissenbasen kann es vorkommen, dass sich Fehler einschleichen. So veraltern Wissenbasen zum Beispiel, wenn sie nicht ständig gewartet werden. Ob eine Wissensbasis in allen Teilen korrekt ist kann nur sehr schwer nachgeprüft werden.
Schneider sieht den Benutzer selbst in der Verantwortung Schaden dadurch zu vermeiden, dass er die gefunden Lösungen anhand der Erklärungskomponente

nachprüft. Da möglicherweise verschiedene Menschen an einer Wissensbasis arbeiten, können sich zudem Fehler einschleichen, die erst sehr spät gefunden werden. So könnte eine verwendete Aussage gleichen Inhalts von verschiedenen KEs verschieden benannt werden (siehe hierzu auch Kapitel 6).
Der Experte besitzt neben seinem Spezialwissen noch über ein Allgemeinwissen, das ihm in verschiedenen Problemkontexten zur Verfügung steht. Dieses kann von einem XPS nicht vollständig abgebildet werden, weshalb viele einfache Fehler, die der Mensch sofort sieht, weil er Erfahrung mit seiner Umwelt besitzt, vom XPS möglicherweise gar nicht gefunden werden.
Wenn der Experte bei seiner Problemlösung an seine Grenzen stößt, bemerkt er dies und hat möglicherweise auch eine Lösung dafür, z. B. bestimmet Dinge zu recherchieren, die entsprechenden Dokumentation wieder mal zur Hand zu nehmen oder sich einfach Hilfe eines anderen Experten in Anspruch zu nehmen. Das XPS stellt hingegen - falls keine Lösung gefunden wurde - die Arbeit einfach ein.

4.1.3 Überblick über die Komponenten eines XPS

Der Aufbau von Expertensystemen in die verschiedenen Komponenten ist schon sehr alt und hat sich - für mich erstaunlicherweise - in den letzten 25 Jahren nicht wesentlich geändert. So wird in der Literatur meist noch immer die von /Puppe 88/ verwendete Darstellung verwendet.
Die zu Beginn des Kapitels aufgestellten Forderungen nach Integration von XPS in die Betriebsumgebung lässt sich aber nur dann einhalten, wenn man neben den klassischen Komponenten noch andere hinzufügt, die spezifische Belange der Produktion berücksichtigen.
In Abbildung 4.2 habe ich neben den klassischen Komponenten auch noch zwei besondere einfügt, die uns innerhalb der Entwicklung von XPS im Bereich technischer Diagnose als sinnvoll erschienen. Die in dieser Darstellung noch klassisch dargestellte Inferenzkomponente erfährt im weiteren Verlauf aber eine drastische Umstrukturierung, da es mir sinnvoll erscheint den Problemlösungsvorgang anpassbar zu machen und deshalb Inferenzschritte als Metawissen der Wissensbasis zuzuordnen.
Zu den dargestellten Komponenten gehört ebenfalls der im Kapitel 4.5 dargestellte Parser, der verwendet wird, um externe und interne Wissensrepräsentation ineinander überzuführen.

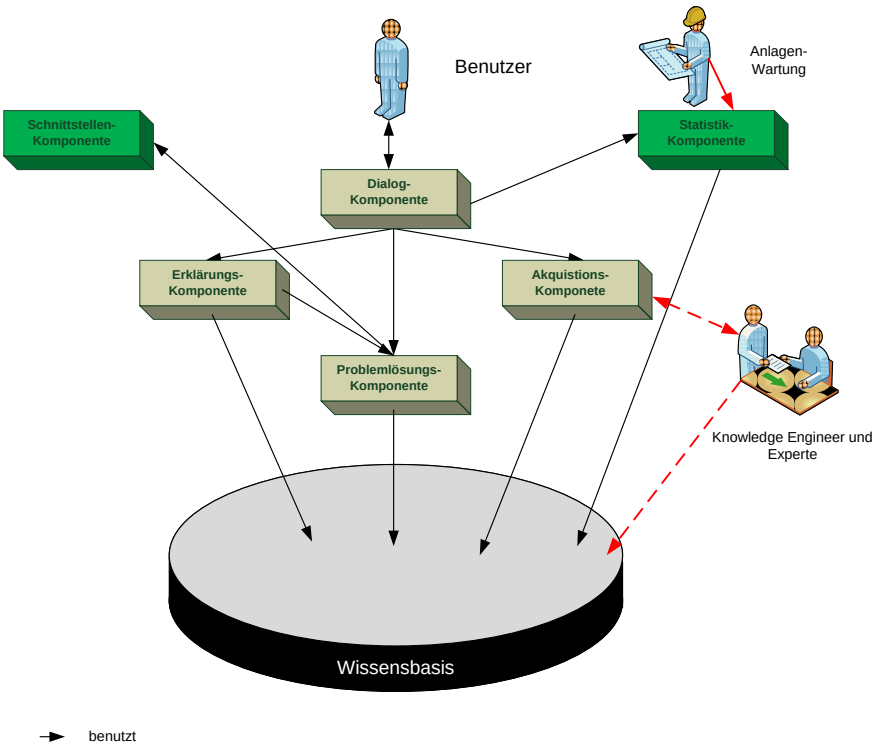


Abbildung 4.2: *Komponenten eines XPS*

Die Aufgaben der einzelnen Komponenten lassen sich dabei kurz wie folgt beschreiben.

Dialog-Komponente
Stellt die Schnittstelle zum Benutzer dar. Sie umfasst dabei verschiedene Oberflächen, die sich an der Tätigkeit des Benutzers orientieren. Beispiele für die verschiedenen Oberflächen werden in Kapitel 4.3 gezeigt.
Inferenz-Komponente
Die Problemlösungs-Komponente oder Inferenzer verwendet das in der Wissensbasis abgespeicherte Wissen, um ein vom Benutzer vorgelegtes

Problem zu lösen. In Kapitel 4.6 werde ich hierzu eine Aufteilung der Komponenten in verschiedene Problemlösungsschritte beschreiben. Dieses Vorgehen ermöglicht es dem KE die Problemlösung sehr diffizil anzupassen.

Erklärungs-Komponente
Die Erklärungs-Komponente beschreibt den eigentlichen Problemlösungsvorgang, so dass der Nutzer den Problemlösungsvorgang des XPS genau nachvollziehen kann. Vor allem im Bereich der medizinischen Diagnose, in der der Arzt vor der Therapie überprüfen möchte, ob die Diagnose ordnungsgemäß erfolgt ist.

Akquisitions-Komponente
Die wohl komplexeste Komponente beschäftigt sich mit der Eingabe von Wissen in die Wissensbasis des XPS. Dabei muss dafür gesorgt werden, dass das Wissen syntaktisch korrekt und vollständig sowie, soweit dies überprüfbar ist, auch semantisch korrekt ist. Da der Benutzer aber sehr viele Freiheiten hat, ist es nicht möglich die Korrektheit der Wissensbasis zu jedem Zeitpunkt sicherzustellen (sieh hierzu Kapitel 4.5).

Wissensbasis
In der Wissensbasis wird das Bereichswissen abgespeichert. Damit das Wissen ordnungsgemäß verarbeitet werden kann ist die vorher gewählte Wissensrepräsentation, sowie eine für das XPS definierte Wissensrepräsentations-Sprache einzuhalten.

Beispielhaft für spezielle zusätzliche Komponenten möchte ich hier zwei weiter Komponenten kurz aufführen, die sich in der technischen Diagnose bewährt haben. Da diese Komponenten auf den speziellen Anwendungsfall angepasst sind, werde ich sie im weiteren Verlauf nicht näher behandeln. Der Entwickler von XPS sollte sich aber mit den entsprechenden Erweiterungen beschäftigen, um das XPS in die entsprechende Betriebsumgebung einpassen zu können.

Schnittstellen- Komponente
Die Komponente wurde von den von mir mitentwickelten XPS zur Abfrage an die Speicherprogrammierbaren Steuerungen (SPS) verwendet. Hierzu musste ein Datenbaustein in die Steuerung integriert werden, der dann bestimmte Überprüfungen vornahm und das XPS starteten sobald ein entsprechendes Datum auftrat. Die Komponente wurde innerhalb des

Problemlösungsvorgangs dazu verwendet Daten aus der Steuerung auszulesen, was den Diagnosevorgang für den Nutzer wesentlich vereinfachte.

Statistik-Komponente
Diese Komponente wurde von uns dazu verwendet bestimmte Austauschteile einer technischen Anlage zu überwachen. Damit konnte zum Beispiel der vorsorgliche Austausch verschiedener Teile initiiert werden, weil hier festgehalten werden konnte, in welchem Zeitintervall bestimmte Teile mit großer Wahrscheinlichkeit ausfallen würden.
Darüber hinaus können, je nach Einsatzgebiet, auch noch andere Komponenten sinnvoll sein. /Gabriel 08/ erweitert diese Struktur zum Beispiel durch eine Lernkomponente und weist darauf hin, dass auch die Anbindung an Datenbanken sinnvoll sein kann.

4.1.2 Shells/Tools

Schon sehr bald innerhalb der Entwicklungsgeschichte von XPS wurde erkannt, dass man zur Anwendung eines Systems auf einen anderen als den bisherigen Problembereich nur die entsprechende Wissensbasis auszutauschen hatte. So entstand als erster Expertensystem-Entwicklungsumgebung EMYCIN. (siehe Kapitel 2.4.1)
Behält man die Wissensrepräsentation und ihre Darstellung im System bei, so muss man nur die Wissensinhalte ändern, da ja die komplette sonstige Verarbeitung durch die bereits realisierten Komponenten durchgeführt werden kann.
Man unterscheidet bei den Entwicklungssystemen zwischen der Shell und einem Toolkit. Während die Shell eine bereits lauffähige Umgebung darstellt, in der nur die Wissensbasis aufgebaut und verändert werden kann, stellt ein Toolkit eine Umgebung dar, in der auch die verschiedene anderen Komponenten auf den Anwendungsfall angepasst werden können. (siehe hierzu auch /Petcu 98/)
Nochmals sei darauf hingewiesen, dass der Aufbau einer Wissensbasis in einer Entwicklungsumgebung aber in der Form einer vorab festgelegten Wissensrepräsentation vorgenommen werden muss, die von den Werkzeugentwicklern festgelegt ist. Dabei werden meist verschiedene Repräsentationen angeboten, so dass etwa reine Regelbasierte Systeme, aber auch hybride Wissensbasen entstehen können.

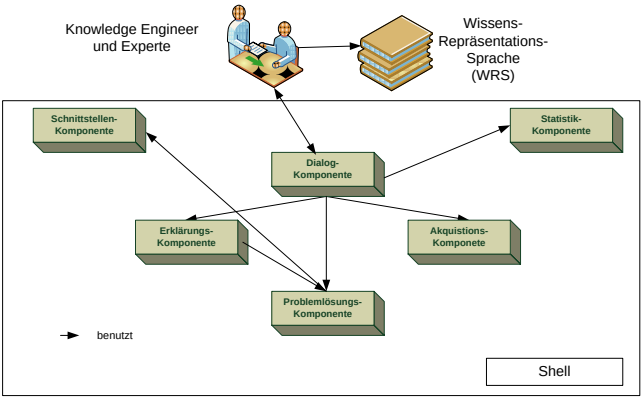


Abbildung 4.3: *XPS-Shell*

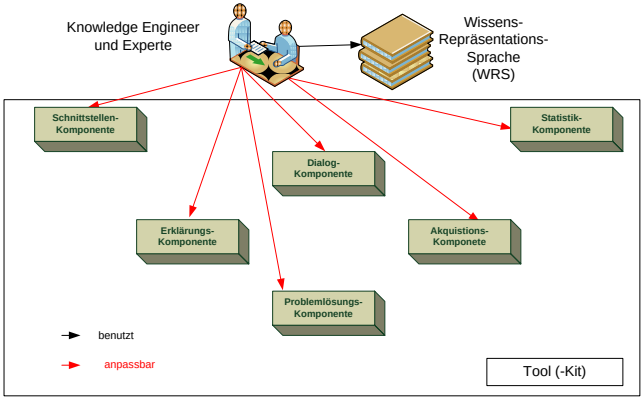


Abbildung 4.4: *XPS-Toolkit*

Zu Beginn der XPS-Entwicklung wurden die Werkzeuge vor allem zur schnellen Erstellung von Prototypen verwendet. Heute wird oft mehr Aufwand auf die Erstellung einen Modells gelegt /Pietzka 95// Wachsmuth 94//Cawsey 03/ (siehe hierzu auch Kapitel 6).

4.2 Die Wissensbasis

In der Wissensbasis wird das Wissen organisiert. /Budin 04/ spricht hier verallgemeinernd von Wissensorganisationssystemen, wozu er auch Thesauri und andere Klassifikationssysteme zählt, die nach seiner Meinung als (extrinsische) Ontologien und (intrinsische) Ontologien genutzt werden können. Zur Einbeziehung von Ontologien in Expertensystemen sei auf Kapitel 6 verwiesen.

Wissensbasen lassen sich nicht nur in XPS verwenden sondern sind auch in Elearning-Systemen oder in Online-Archiven oder Content-Management-Systemen verwendbar. Die verschiedenen Systeme können heute miteinander vernetzt werden. Prinzipiell lässt sich über das Web auch hierauf zugreifen. Wichtig scheint mir in diesem Zusammenhang aber zu sein, dass hierzu eine allen Beteiligten bekannte und verarbeitbare Wissensrepräsentations- und –Zugriffssprache vorliegen muss.

Im Bereich der XPS wird in der Wissensbasis das Bereichs-Wissen abgespeichert. Es wird dazu in einer Repräsentationsform dargestellt und den anderen Komponenten über Schnittstellen zur Verarbeitung bereitgestellt. Die in der Künstlichen Intelligenz im Bereich der XPS gebräuchlichsten Repräsentationsformen sind dabei Regeln und Frames. Die Frames können ihrerseits wieder in verschiedenen Objektnetzen angeordnet werden. Grundsätzlich sind auch andere Repräsentationsformen wie Entscheidung-Bäume und –Tabellen /Bohrer 04/, Semantische und Neuronale Netze oder Constraints denkbar. Die Repräsentationsform sollte sich dabei am abzubildenden Wissen ausrichten. Prinzipiell sind die meisten Formen aber in die Regel/Objektrepräsentation überführbar (siehe hierzu auch /Hartmann 09/).
Wird mehr als eine Repräsentationsform zur Wissensabbildung verwendet, spricht man von einer hybriden Darstellungsform. Weit verbreitet ist die Kombination von Objekten und Regeln, wobei die Objekte zum Beispiel in der technischen Diagnostik über entsprechende Relationen, z.B. „ist-Teil-von“ und „besteht-aus“ Baugruppen und die dazugehörigen Bauteile abbilden.

Die ersten Expertensysteme waren rein Regelbasiert, was dazu führte, dass tausende von Regeln verarbeitet werden musste. Durch die Einführung von Objekten und die Zuordnung der Regeln zu den verschiedenen Bauteilen ergab sich in der technischen Diagnostik eine Strukturierung, die eine einfachere Modellierung möglich machte. In Kapitel 5 „Rotes Wunder“ wird ohne Einschränkung der Allgemeinheit eine beispielhafte Wissensrepräsentation dargestellt.

4.2.1 Grundsätzliche Bemerkungen zu Regeln und Frames

Mit Regeln können Zusammenhänge in Form eines WENN…DANN-Konstrukts beschrieben werden. Damit hat man die Möglichkeit Reiz/Reaktion-, Ursache/Wirkung- oder Symptom/Diagnose-Zusammenhänge abzubilden. Aufgrund des einfachen und für den Menschen direkt verständlichen Gebrauchs findet man einen regelbasierten Anteil auch heute noch in vielen KI-Anwendungen.
Beispiel:

WENN	(Literal ODER Literal) UND NICHT Literal	DANN	**Literal UND Literal**
	PRÄMISSE		*KONKLUSION*

In Anlehnung an die Logik wird auch eine Regel in zwei Teile unterteilt, die man Prämisse und Konklusion nennt. Die Prämisse besteht aus einer Reihe von negierten oder nicht negierten Wissensbestandteilen, die mit UND (Konjunktion) oder ODER (Disjunktion) miteinander verbunden sind. Diese als Literale bezeichneten Elemente können dabei zum Beispiel folgenden Aufbau haben (Beispiele in Klammern):

- eine einfache Aussage, etwa in Form einer aus der Aussagenlogik bekannten Aussage, die wahr oder falsch sein kann („Drehzahl_zu_hoch“),
- die Überprüfung der Wertebelegung einer Aussage („Farbe_des_Öls“ = „farblos“)
- ein Vergleich zwischen zwei Aussagen („Strom_an_Schalter1“ = „ Strom_an_Schalter2“)
- Aufruf einer Funktion, die ebenfalls wahr oder falsch liefert (Speicherstelle (Nr))
- die Überprüfung des Rückgabewertes einer Funktion (Speicherstelle (Nr) = 1)

- ein Vergleich zwischen zwei Funktionsaufrufen (Speicherstelle (4711) = Speicherstelle (0815)).

Literale der Konklusion werden dabei meistens verwendet als:

- Einfache Aussagebelegungen („Motor1_defekt“), die Aussage wird vom System dann mit wahr belegt.
- Wertzuweisung. Einer Aussage wird ein spezieller Wert zugewiesen („Motor_an_Station1“ = „defekt“).
- eine Aktion. Etwa DIAGNOSE „Der Motor der Station1 ist defekt.! Bitte wechseln Sie ihn aus!“.

Die Literale der Konklusion können ebenfalls mit UND verbunden werden. Dies wird hier aber im Sinne einer sequentiellen Verarbeitung verwendet.
Statt UND, ODER und NICHT können hier natürlich auch andere Symbole etwa AND, OR, Not, „+“, “-„ … verwendet werden. Auch das hier beispielhaft verwendete Zuweisungszeichen „=“ kann natürlich in verschiedener Form vorkommen.
Damit die Aussagen von einem XPS überhaupt erfragt und überprüft werden können müssen sie neben ihrem Namen, über den sie angesprochen werden können, entsprechenden Bestandteile besitzen:

- einen Wertebereich. Diese legt fest welche Werte überhaupt zugeordnet werden können. Beispiele sind hier die hinlänglich bekannten INTEGR, BOOL, REAL, aber auch Aufzählung etwa („farblos“ „braun“ „honigfarben“ „schwarz“) oder Intervall (etwa [1..100]).
- einen Wert. Dieser hält dann den Wert des Wertebereiches fest, der vom Benutzer eingegeben oder vom System ermittelt wurde.
- Einen Fragetext. Mit diesem wird der Benutzer nach dem Wert gefragt.
- einen Antwort/Diagnosetext, der bei Auftreten in der Konklusion einer Regel ausgegeben werden kann.
- Eventuell die Referenz zu Medien wie Bilder, Videos oder Erklärungstexten.
- einen Auswahlanzeiger. Mit dessen Hilfe können Teilmengenbeziehungen verarbeitet werden. Der Anzeiger wird zum Beispiel als „N aus M“ –bezeichnet. Ist er gesetzt kann der Nutzer mehrere Elemente aus einem Wertebereich wählen, ist er nicht gesetzt nur jeweils einen Wert. Innerhalb einer Prämisse können dann auch Literale der folgenden Art verwendet werden: „Wochentag aus

(Mo, Di, Do)“, wobei hier mehrere Tage auf einmal ausgewählt werden können.

Um die Regeln innerhalb eines Systems gezielter verarbeiten zu können, werden diese zu Regelgruppen zusammengefasst, die ebenso wie die Regeln selbst benannt werden können. Dadurch lassen sie diese Regelgruppen bzw. Regeln von verschiedenen Stellen aus ansprechen (siehe hierzu auch Anhang B). Eine konkrete Wissensrepräsentation wird in Kapitel 5 dargestellt.

4.2.2 Frames

Die von Minsky zur Beurteilung von Szenen und Dialogen 1975 eingeführten Frames (engl. Rahmen) stellen eine weitere hier kurz vorgestellte Art der Wissensrepräsentation dar. Sie sind die Vorgänger der heute bekannten Klassen innerhalb der Objektorientierten Programmierung. Dabei bildet ein Frame eine ganze Klasse von Objekten und kein einzelnes reales Objekt ab. Von dieser Klasse werden dann erst zur Laufzeit reale Objekte abgeleitet. Diesen Vorgang nennt man in der KI Instanziierung oder Spezialisierung (siehe hierzu auch Spezialisierung-Generalisierung, Kapitel 5), das erhaltende Objekt Instanz./Barr 81-82/
Generell setzt sich ein Frame aus verschiedenen Slots (Einschübe) zusammen. Den Slots werden dabei verschiedene Bedeutungen zugeordnet, wobei einige bereits spezielle Bedeutungen besitzen während andere erst vom Ersteller der Wissensrepräsentation festgelegt werden. Die Anzahl der Slots ist prinzipiell nicht begrenzt.
Die Slotnamen innerhalb einer Instanz sowie Vorbelegungen werden von den Klassen übernommen, man spricht dabei von Vererbung. Werte werden dem Objekt während der Instanziierung oder innerhalb des Systemlaufs zugeordnet. Innerhalb der Problemlösung eines XPS werden die Werte einer Aussage zum Beispiel durch Befragung des Benutzers belegt, der Wertebereich während der Instanziierung übernommen (sie auch 4.2.1).

/Barr 81-82/ haben den Aufbau eines Slots beispielhaft so dargestellt:

Slotname:	Wert
	Wertebereich (z.B. integer)
	optional DEFAULT (z.B. 3)
	Belegungsprozedur (z. B. if-needed)
	Prozeduren/Funktionen
	Liste

Rein Objektorientierte Repräsentationsformen machen Prozeduren notwendig, die es dem Benutzer ermöglich die Slots der Instanzen zu belegen. Bemerkt also ein Problemlöser, dass verschiedene Slots nicht belegt sind, kann er die entsprechenden Prozeduren ausführen sobald der entsprechend Slot belegt oder verändert werden soll. Die verwendeten Prozeduren werden auch als Trigger bezeichnet /Barr 81-82/.
Es ist sinnvoll in den Klassen bereits Wertebereiche wie String, Integer oder Liste festzulegen, auch wenn dies nicht geschehen muss. Der Wert NIL lässt dem Benutzer zum Beispiel die Wahl der Belegung völlig frei. Im Sinne einer korrekten Überprüfung scheint dies innerhalb der Wissensrepräsentation eines XPS heute nicht mehr sinnvoll.

Beispiel:

KLASSEN-NAME	Automobil
INSTANZEN-NAME	STRING
BESTEHT_AUS	Liste
PS	Integer
ERSTZULASSUNG	Datum
BESITZER	String

Instanz

KLASSEN-NAME	Automobil
INSTANZEN-NAME	Mercedes-Benz SLK
BESTEHT_AUS	(Karosserie Motor)
PS	200??
ERSTZULASSUNG	1.1.2012
BESITZER	K. Hartmann (leider nur Wunschvorstellung)

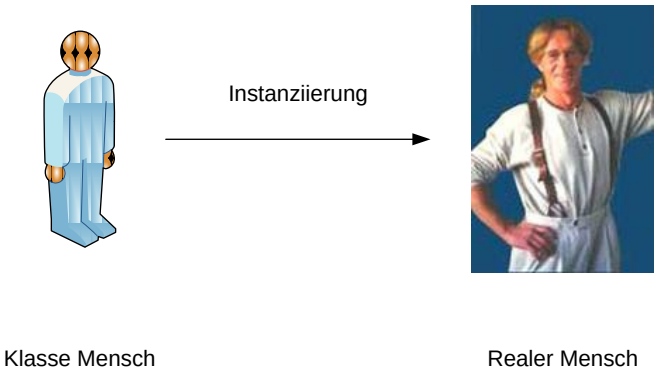


Abbildung 4.5: *Beispiel einer Klasse und einer daraus abgeleiteten Instanz*

Über speziellen Slots, etwa „Besteht-aus“ und „Ist-Teil-von“ benannt, können ganze Netzwerke von Objekten dargestellt werden. Diese werden zum Beispiel in der technischen Diagnose dazu verwendet, um Bauteilebäume (Baugruppen, Bauteile und deren Bestandteile) darzustellen (siehe hierzu auch Kapitel 5 und Anhang B).
Mit dem nur in der KI verwendeten Slot BEHAVIOUR beschreibt man das normale Verhalten einer Instanz. Bindet man an diesen Slot eine Prozedur können damit zum Beispiel beim Instanziieren Werte belegt oder inferiert werden.
Bei der Instanziierung werden alle Slotbelegungen an die Instanz weitergegeben (vererbt). Damit gelten alle in einer Klasse definierten Slots zunächst für alle abgeleiteten Instanzen. Bei der Beschreibung realer Objekte werden auch Eigenschaften der Objekte durch Slots festgelegt. Es gibt nun aber Eigenschaften, die nicht zu allen abgeleiteten realen Objekten gehören. In der KI wird an dieser Stelle oft Tweety, der Vogel genannte, der nicht fliegen kann (möglicherweise ein Strauß oder Pinguin, wahrscheinlich aber ein Hinweis auf eine spezielle Zeichentrick-Figur).
Die Eigenschaft „Kann-Fliegen“, die für andere Vögel gilt, muss also für Tweety ausgeschlossen werden. Bei der Instanziierung wird dann zum Beispiel der entsprechende Slot (Eigenschaft) auf NIL (Ausschluss der Vererbung) gesetzt.

Weitere Möglichkeiten zur Einbeziehung von Objekten in Wissensbasierten Systemen werden z. B. in /Marchand 85/ aufgeführt.
Generell ist die Implementierung von Frames mit jeder Programmiersprache möglich, die Strukturen oder Records zulässt.
Es sei darauf hingewiesen, dass die hier dargestellten Frames nur Verwandte der in der OOP verwendeten Klassen und Instanzen sind, selbst wenn sie in der internen Verarbeitung von XPS wie Klassen und Instanzen verwendet werden. Eine Detaillierte Beschreibung der Unterschiede würde hier den Rahmen sprengen. (Siehe hierzu auch /Lahr 09/)

4.2.3 Hybride Repräsentationsformen

Unter Hybriden Repräsentationsformen versteht man hier die Verbindung zwischen verschiedenen elementaren Formen, so werden Frames und Regeln gerne miteinander verbunden. Die Regeln können dann zum Beispiel einfach über einen Slotnamen angesprochen werden.

Beispiel:

KLASSEN-NAME	Automobil
INSTANZEN-NAME	..
..	..
DIAGNOSEREGEL	Regel1
REGELGRUPPE	(Regel1 Regel2 …)
REGELGRUPPENNAME	RG-Motor-Probleme

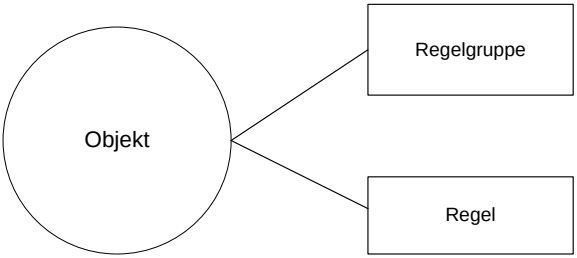


Abbildung 4.6: *Anbindung der Regeln an die Objekte*

Regeln können entweder direkt als zu verarbeitende Liste oder über ihren Namen (etwa Regel1oder RG-Motor-Probleme) angesprochen werden. Entsprechend der implementierten Verarbeitungsstrategie werden sie berücksichtigt sobald das Objekt in der Verarbeitung „angefasst" wird. Die Regelverarbeitung erfolgt dann wie innerhalb einer normalen rein regelbasierten Repräsentation (siehe hierzu 4.6 Problemlösungskomponente). Regelgruppe stellen hier nur Zusammenfassungen von Regeln dar.

4.2.4 externe und interne Repräsentation

Das in einer Wissensbasis in einer bestimmten Repräsentationsform vorliegende Wissen, wird von einem Computerprogramm verarbeitet werden müssen. Die Abbildung dieses Wissens wird heute dabei in einer Form geschehen, dass Sie objektorientiert verarbeitet werden kann. Diese Form ist aber vom Menschen nicht leicht lesbar, zumal man davon ausgehen muss, dass der Experte kein Programmierer ist, sondern sein Spezialwissen auf anderem Gebiet liegen sollte.
Die Entwickler tun gut dran, wenn sie sich bei der Darstellung des Wissens einer Form bedienen, die vom Experten leicht lesbar ist und die auch als Teil des mentalen Modells verstanden werden kann (siehe hierzu Kapitel 6).
Aus diesen Gründen unterscheidet man meistens eine externe und interne Repräsentationsform. Innerhalb des Knowledge Engineering wird in Zusammenarbeit mit dem Experten eine Wissensrepräsentations-Sprache (WRS) entwickelt, die sich aus dem abzubildenden Wissen und dem Problemfall ergibt und vom Experten auch verstanden wird.
Diese externe Repräsentation kann mit einem einfachen Texteditor verarbeitet werden und ist ähnlich einer Programmiersprache aufgebaut. Die Wissensbasis kann in dieser Form vom System eingelesen werden und wird beim Einlesen durch einen Parser in die vom Computerprogramm verarbeitbare interne Repräsentation umgewandelt. Aus den verschiedenen Repäsentations-Konstrukten werden dabei Objekte im Sinne der OOP gestaltet, auf denen dann der Problemlösungsvorgang ablaufen kann.
Die von der Akquisitionskomponente erstellten oder veränderten Teile der Wissensbasis werden vom Parser beim Speichern der Wissensbasis wieder von der internen in eine externe Form gewandelt.
Der Vorteil der Verwendung eines Parsers liegt dabei in der lesbaren Form einer externen Präsentation, die auch von einem nicht direkt befassten Manager verstehbar ist und als Dokumentation dienen kann. Da heute fast ausschließlich XPS-Werkzeuge verwendet werden, muss die Wissensbasis erst

geladen werden und besitzt schon aus diesem Grunde eine externe Repräsentation. Ein Kriterium, welches Werkzeug verwendet werden soll, kann es sein ob diese Form auch lesbar und damit weiterverwendbar ist.
In Kapitel 5 wird im Rahmen des Projekts „Rotes Wunder“ eine Wissensrepräsentationssprache vorgestellt, die im Rahmen der Vorlesung „Wissensbasierte Systeme“ entwickelt wurde. (siehe Anhang B)

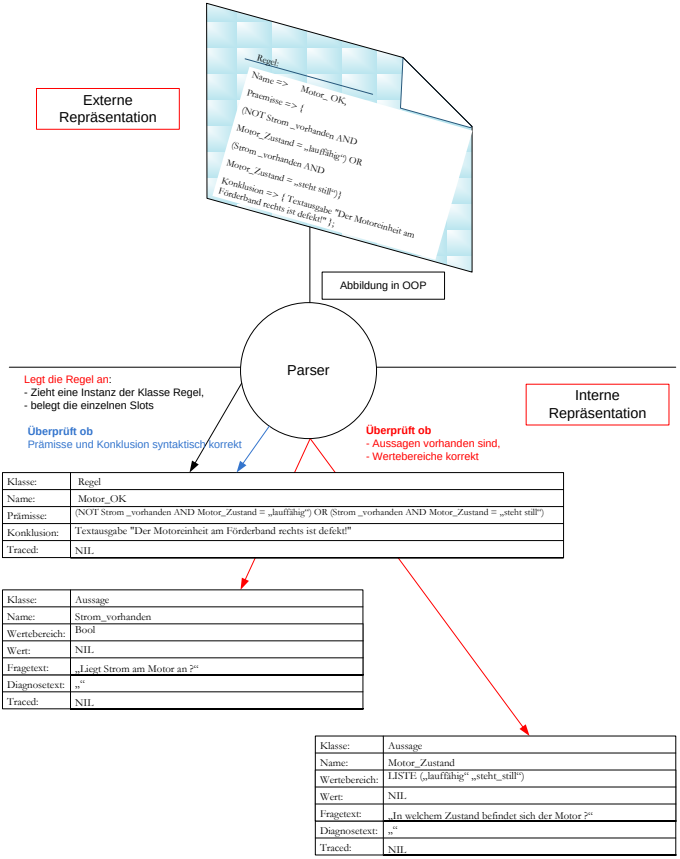


Abbildung 4.7: *Überführung von interner zur externer Repräsentation*

4.3 Die Dialogkomponente

Innerhalb eines XPS können verschiedene Dialogmodi unterschiede werden. So identifiziert /Wischnewsky 10/Modi, die ich wie folgt zusammenfassen möchte:

- Handbuch- und Beratungsmodus,
- Wissenspflege- und Notizbuchmodus und
- Experimentier- und Kommentar- bzw. Erklärungsmodus.

Der Handbuchmodus verwendet die Wissensbasis eines XPS als Nachschlagewerk. Der Benutzer kann sich dabei zum Beispiel den Aufbau einer technischen Anlage vergegenwärtigen oder sich einarbeiten (das XPS als Tutor).
Im Gegensatz hierzu wird das XPS im Beratungsmodus seiner eigentlichen Aufgabe nachkommen und den Benutzer bei der Problemlösung unterstützen.
Wissenspflegemodus und Notizbuchfunktion werden zur Pflege der Wissensbasis verwendet. Der Wissenspflegemodus wird dazu verwendet, um:

- die Wissensbasis fehlerfrei aufzubauen,
- Fehler, die beim Laden einer Wissensbasis gefunden werden, zu eleminieren,
- die Wissensbasis zu tunen, in dem zum Beispiel Frage- und Diagnose-Texte angepasst werden.

Die Notizbuchfunktion wird hingegen dazu verwendet, um dem Benutzer die Möglichkeit zu geben seine Gedanken schnell einbringen zu können, ohne den eigentlichen Ablauf des Systems zu unterbrechen. So wird zum Beispiel der momentane Zustand der Wissensbasis mit allen Belegungen abgespeichert werden können, um den KE zu ermöglichen einen ganz speziellen Zustand zu simulieren.
Der Experimentiermodus setzt auf einem bestimmten Zustand einer Wissensbasis auf. Um die Wissensbasis überprüfen zu können, muss in einer Simulation verschiedener Aussagenbelegungen und Problemlösungsverläufe durchgeführt werden können. Das Testen kann dabei manuell oder teilweise auch automatisch erfolgen. Der KE kann sich dabei durch die Wissensbasis „durchtesten“, wobei er immer wieder die Notizbuchfunktion benutzt, um sich bestimmte Zustände zu merken. Das automatische Testen kann zum Beispiel beim Überprüfen der verschiedenen Problemlösungsmechanismen vorgenommen werden. Damit können zum Beispiel fehlende Texte oder ein fehlerhafter Problemlösungsverlauf gefunden werden, die durch Optionen

innerhalb der Wissensrepräsentationen auftreten können. Zum Beispiel könnte eine Aussage vom KE nur innerhalb der Prämisse einer Regel vorgesehen worden sein, kommt aber doch auch nochmal in einer Konklusion vor, hat dafür aber keinen Text, der dem Benutzer ausgestellt werden könnte.
Eine der elementaren Anforderungen, die an eine XPS gestellt werden, ist es dem Benutzer die Problemlösung zu erklären. Dazu werden zum Beispiel die Ein- und Ausgaben, sowie der Zustand der Wissensbasis vermerkt und bei Bedarf dem Benutzer dargestellt. Dieser Erklärungsmodus wird in der Experimentierphase sehr oft vom KE verwendet.
Prinzipiell sollten die Modi vom Benutzer frei wählbar sein. Allerdings sollte man dafür sorgen, dass nicht jeder Nutzer zu allen Features des Systems Zugriff hat. So sollte die Akquisitionskomponente wirklich nur von den entsprechenden Experten und KEs aufrufbar sein, um eine ungewollte Veränderung durch einen weniger versierten Benutzer zu vermeiden.

Auf die auch in der Literatur genannten Chatbots wird hier nicht eingegangen, da sie eher ein-Ausgabesystem als XPS darstellen. /Dabringer 09/ charakterisiert sie als textbasierte Dialogsysteme mit Texteingabe- und Ausgabemasken, über die sich in „natürlicher“ Sprache mit dem System kommunizieren lässt. Eliza von Weizenbaum stellt ein solches System dar (siehe hierzu auch Kapitel 2).

4.3.1 Einstieg und prinzipieller Aufbau einer Oberfläche

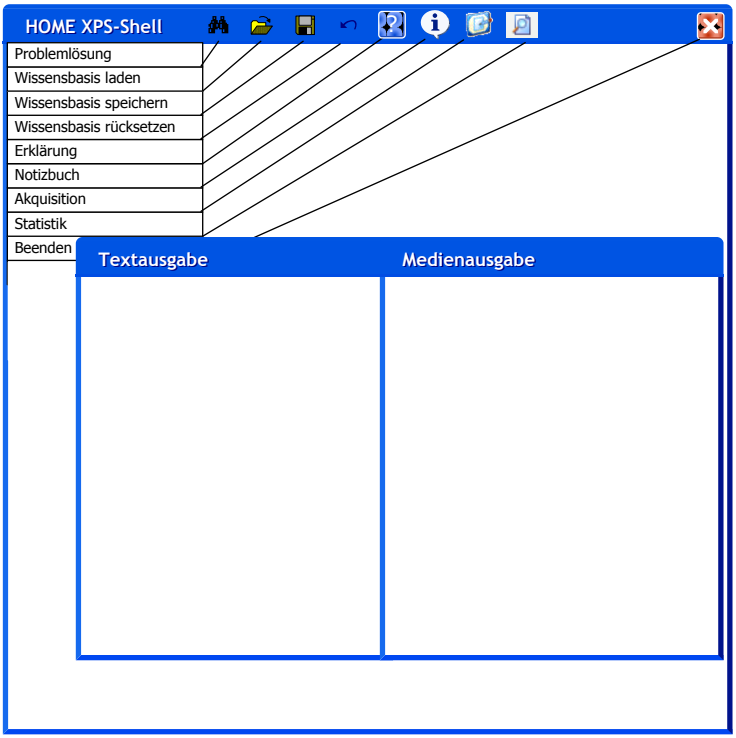


Abbildung 4.8: *Einstiegsbildschirm mit Medien und Textbereich zufügen.*

Über den Einstiegsbildschirm wird es dem Benutzer ermöglich verschiedene Modi zu wählen. Das oben angeführte Beispiel zeigt die Funktionalität, die mir hierbei sinnvoll erscheint, ohne die Anordnung als bindend zu betrachten. Der Button Statistik sollte hier als Zugang zu weiteren, den klassischen Aufbau eines XPS erweiternden, Komponenten verstanden werden.

Es hat sich als vorteilhaft erwiesen, den Bildschirm in verschiedene Bereiche einzuteilen, die in den verschiedenen Modi gleichartig verwendet werden. So wird einen bestimmter Bereich verwenden, in dem Medien ausgestellt werden, ein anderen in dem Texte dargestellt werden usw. Dies erleichtert es den verschiedene Benutzern die Handhabung des Systems zu erlernen.
Die Anordnung der Buttons sollte dabei so vorgenommen werden, dass sie in den verschiedenen Modi vom Benutzer gleichartig verwendet werden können. So sollte ein Benutzer in jeder Situation das System verlassen können, die Notizbuchfunktion aufrufen und den Zustand der Wissensbasis retten können, um deren Zustand mit dem Experten zu besprechen.

4.3.2 Problemlösungsverlauf

Innerhalb der Problemlösung werden dem Benutzer Fragen gestellt und es werden Lösungen ausgegeben. Um die verschiedenen in Kapitel 4.6 dargestellten Verarbeitung der Regeln und/oder Objekten zu ermöglichen, müssen folgende Eingaben möglich sein:

- Text- oder Zahleneingabe über Tastatur
- Auswahl über Maus oder Pointer, wobei man zwischen Einfach- und Mehrfachauswahl unterscheiden muss.

Werden die Daten direkt eingegeben, muss das System eine Konsistenzprüfung des Wertes vornehmen, weshalb man lieber Auswahlexte verwendet.
Als Ausgaben werden sowohl Texte (auch gesprochene), wie Bilder, kurze Videos oder Animationen verwendet. Es sei darauf hingewiesen, dass die Erstellung von Animationen meist sehr aufwendig ist und diese deshalb eher sparsam eingesetzt werden sollten. Die Medien werden dabei den Aussagen und den Objekten direkt mit einem eigenen Sprachkonstrukt in der WRS zugeordnet (siehe hierzu auch Angang B).
Die Medien werden innerhalb des Problemlösungsvorgangs als Erklärungsmöglichkeit für den Benutzer verwendet, wo eine rein textliche Darstellung zu umfangreich werden würde. Es gilt das Prinzip „ein Bild sagt mehr als tausend Worte“. In meiner Praxis wurde ich einmal auch mit der Erstellung eines Systems konfrontiert, dass völlig ohne Text auskommen sollte, weil die Benutzer kein English konnten (Einsatzgebiet des Systems war die USA).

In den unten aufgeführten Beispielen habe ich einige typische Bildschirmeinhalte während der Problemlösung dargestellt:

- Auswahl eines Objektes, etwa um eine Lösungssuche an dieser Stelle zu Starten oder fortzusetzen
- Eingabe eines Textes zur Belegung einer Aussage,
- Einfach-Auswahl eines Wertes zur Belegung einer Aussage,
- Mehrfach-Auswahl eines Wertes zur Belegung einer Aussage,
- Ausgabe eines Textes, etwa einer Diagnose oder einer Hilfestellung für den Benutzer.

Bei der Ausgabe werden normalerweise bestimmte Bereich vorgesehen, die auf dem Bildschirm hierfür vorgesehen sind, so werden Fragetexte und Medien immer an der gleichen Stelle ausgestellt. Dieses Konzept kann sogar während der Akquisition angewendet werden, wo dem Benutzer dann die entsprechenden Bereiche zur Eingabe von Texten oder zur Einbindung von Medien zur Verfügung gestellt werden, damit dieser direkt beurteilen kann, wie sich dies dem Benutzer darstellen wird.

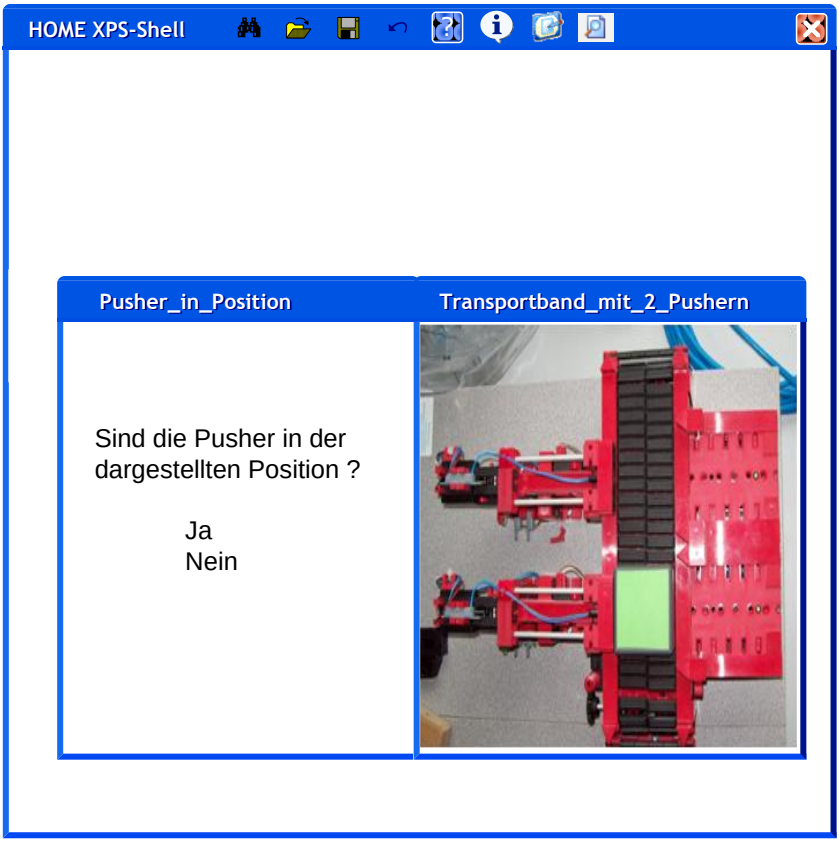


Abbildung 4.9: *Eingabe eines Wertes*

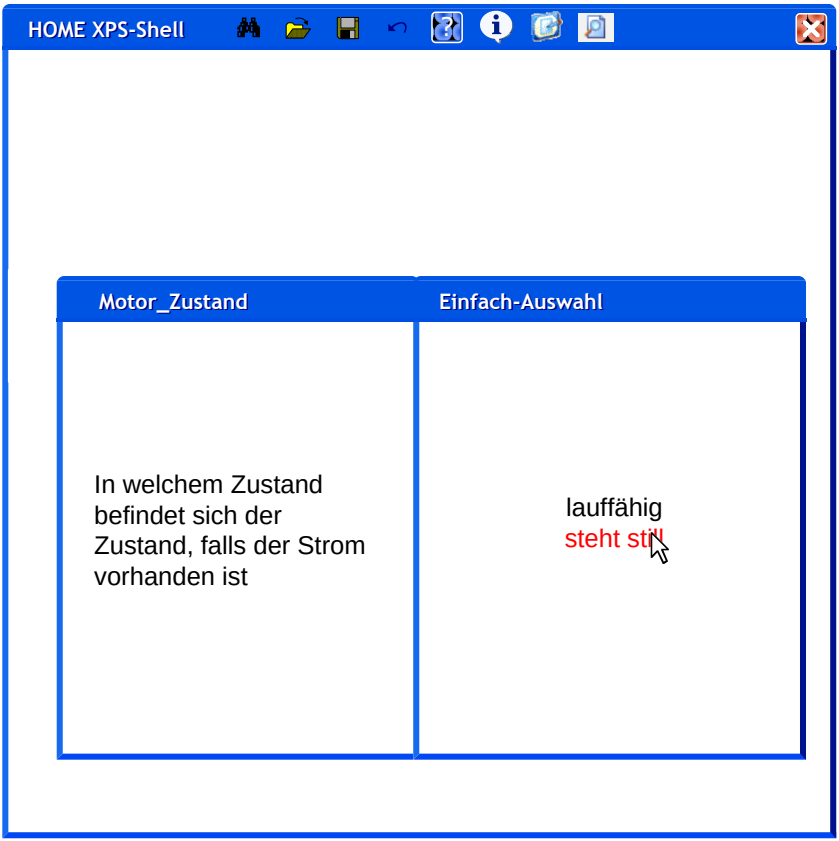


Abbildung 4.10: *Einfach- Auswahl aus einem Wertebereich*

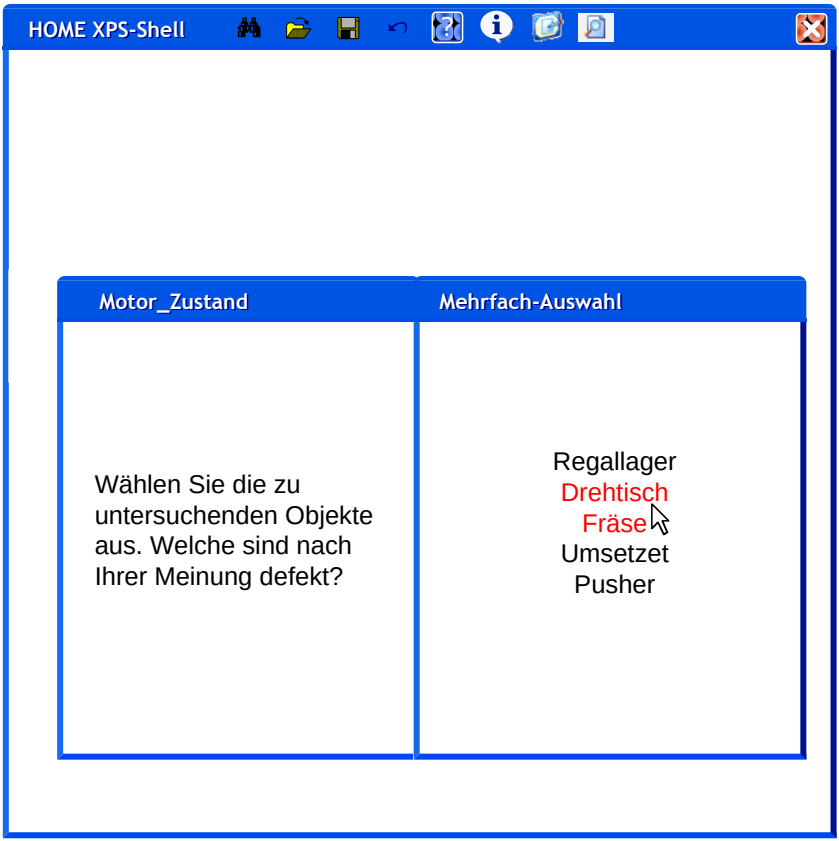


Abbildung 4.11: *Mehrfach-Auswahl aus einem Wertebereich*

HOME XPS-Shell
Diagnose Näherungs_Schalter_r
Induktiver_Näherungsschalter_rechts
Der induktive Näherungsschalter beim rechten Transportband des Umsetzers ist defekt !

Abbildung 4.12: *Diagnoseausgabe*

4.3.3 Erklärung

Innerhalb des Erklärungsmodus wird dem Benutzer der Verlauf der Problemlösung dargestellt oder das System erklärt dem Benutzer sein Verhalten. Dabei müssen ihm seine früheren Eingaben, die über die verschiedenen Schnittstellen erfolgten Belegungen (etwa durch Zugriff auf eine Steuerung) und vom System erzeugten Schlussfolgerungen in sinnvoller Weise dargestellt werden.

Dabei kann man verschiedene Darstellungsmöglichkeiten verwenden:

- die zeitliche Reihenfolge des Problemlösungsvorgangs.

  Diese zeigt den Benutzer zwar das zeitliche Geschehen, zum Beispiel die genaue Folge von Geschehnissen, die zur Lösung oder zur betrachteten Situation führte, macht aber einen ständigen Kontextwechsel notwendig. So führt eine Eingabe des Benutzers etwa zur Auswahl eines Objektes, dem Benutzer wird also nach der Regeldarstellung den Bauteilebaum gezeigt und danach vielleicht wieder, die von diesem Objekt verwendeten Regeln. Erfahrene Benutzer bevorzugen diese Darstellung, während unerfahrene hierdurch leicht überfordert werden können. (siehe Abbildung 4.13).

- die Darstellung eines bestimmten Kontextes.

  Man bleibt bei der Erklärung zum Beispiel solange bei den verarbeiteten Regeln, bis der Benutzer einen anderen Kontext auswählt. Der Benutzer kann sich dabei in aller Ruhe alle angefassten Regeln anschauen, wobei auch die Literale dargestellt werden, die aufgrund der vorgenommenen Eingaben nicht angefasst wurden. Diese Darstellung kann dem KE besonders dann helfen, wenn sich das System anders als erwartet verhält (siehe Bild 4.14).

Das System kann sein Verhalten über einen Kommentar, die vom Benutzer anwählbar sind, weiter erklären. Durch die eingeblendeten Kommentare und mediale Hilfestellungen kann dem Benutzer dargestellt werden, was er in dieser Situation zu tun hat. So kann das System etwa eine bestimmte

Überprüfungsmöglichkeit eines Maschinenteils näher erläutern, wenn nach dem Zustand innerhalb einer Regel gefragt wurde. (sieh herzu Bild 4.15)
Die verschiedenen Erklärungen wurden früher mit verschiedenen W-Fragen (Wie, Warum, Warum nicht, Was ist, Was wäre wenn /Wollny 03, S.40/) verbunden. So konnte man etwa Warum fragen, wenn man wissen wollte, warum das System jetzt genau diese Frage stellte. So konnte zum Beispiel der zeitliche Verlauf der Problemlösung durch wiederholte Warum-Frage dargestellt werden. Wie-Fragen konnten Hilfestellungen ausstellen.
Durch die modernen Darstellungsmöglichkeiten lassen sich heute größere Zusammenhänge direkt darstellen. So kann die Erklärung den zeitlichen Verlauf etwa direkt anhand eines Eingabe- und Zuweisungsbaums darstellen. Hier wird die Problemlösung in Form eines Netzwerkes dargestellt, bei dem der Benutzer eine bestimmte Komponente direkt auswählen kann (siehe Bild 4.14). Statt der W-Fragen verwendet man heute aussagekräftige Icons.

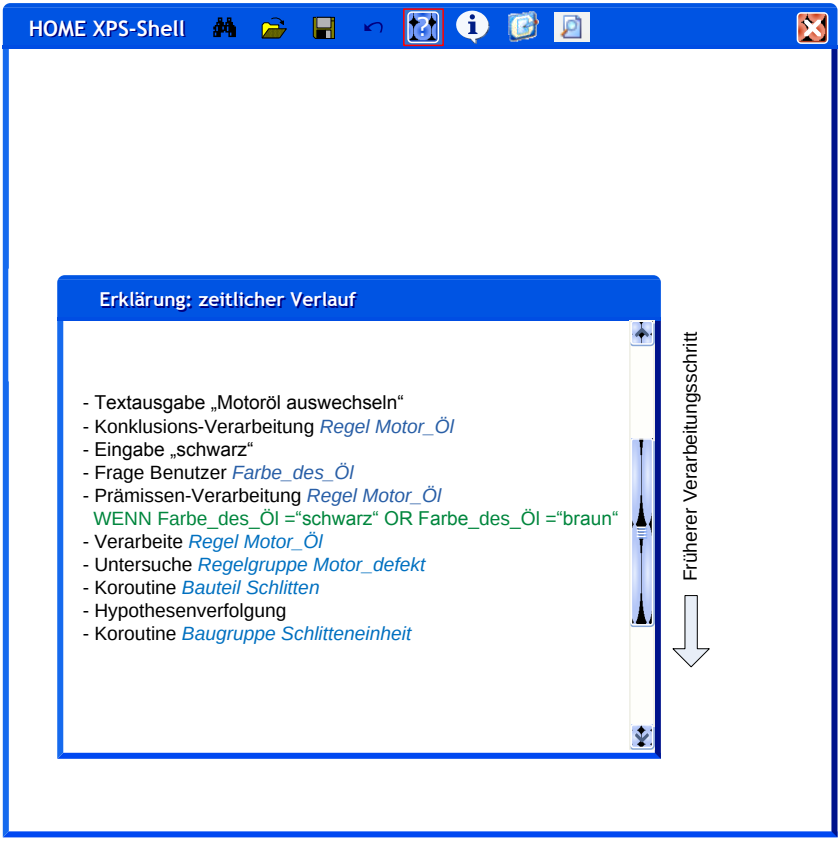


Abbildung 4.13: *Darstellung des Zeitlichen Verlaufs einer Problemlösung*

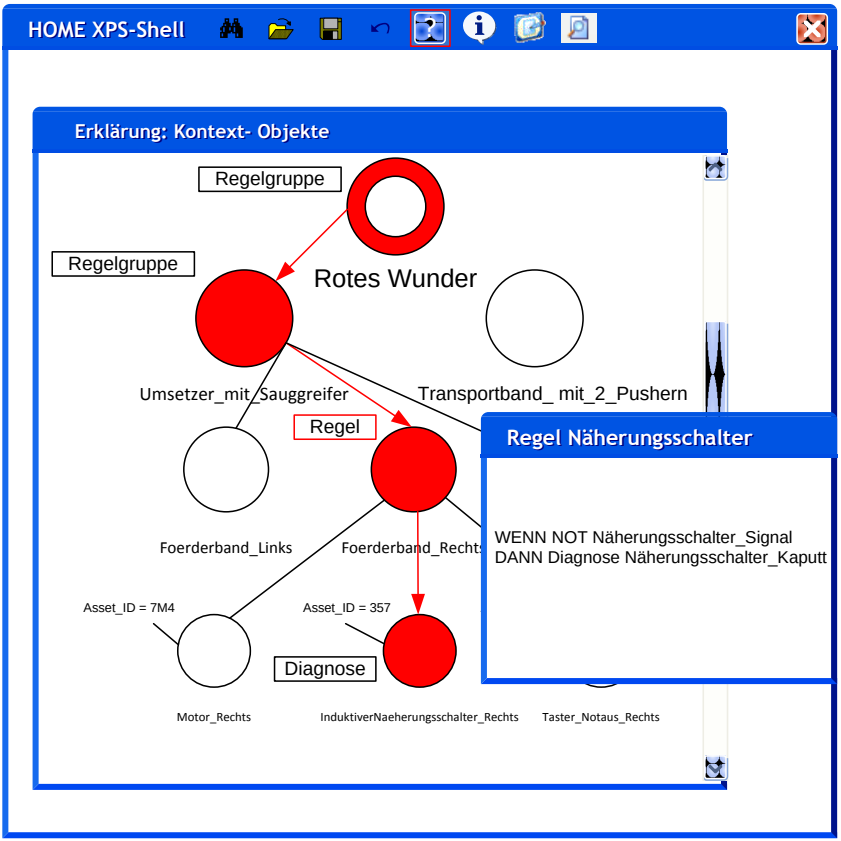


Abbildung 4.14: *Kontext Darstellung hier z.B. die bearbeiteten Regeln*

HOME XPS-Shell
Diagnose Näherungs_Schalter_r
Induktiver_Näherungsschalter_rechts
Der induktive Näherungsschalter beim rechten Transportband des Umsetzers ist defekt! Tauschen Sie ihn aus!
Hilfe zur Diagnose
Zum Austausch des Induktionsschalters schalten Sie zunächst die Stromversorgung der Maschine aus, lösen Sie dann die Verdrahtung, in dem Sie wie folgt vorgehen:
- den blauen Draht an der Klemme A (siehe Dokumentation)

Abbildung 4.15: *Ausstellen von Hilfstexten/ Kommentaren*

4.3.4 XPS als Handbuch

Um ein XPS als Handbuch zu verwenden, muss die Wissensbasis durch zusätzliche erklärende Kommentare ergänzt werden. Im Bereich der technischen Diagnose wird man in der Wissensbasis zum Beispiel die Baugruppen und Bauteile abbilden. Die Zusatzinformationen stellen dann den Verwendungszweck, Ein- und Ausbauinformationen und technische Kenndaten dar.
Der Aufwand die Wissensbasis so zu ergänzen ist nur vertretbar, wenn diese Informationen leicht zugänglich sind, etwa in Form von schon existierenden Handbüchern, die man digitalisieren kann. Machen sie aber eine Wissensaufbereitung notwendig wird der Aufwand sehr schnell zu groß, weil hier ja nur ein Nebeneffekt erzielt werden soll und auch dieser Bereiche eines Systems ständig gepflegt werden muss. Die Hauptanwendung des XPS ist die Problemlösung, zur Ausbildung dienen hingegen Tutorielle Systeme, die etwas anders aufgebaut sind.

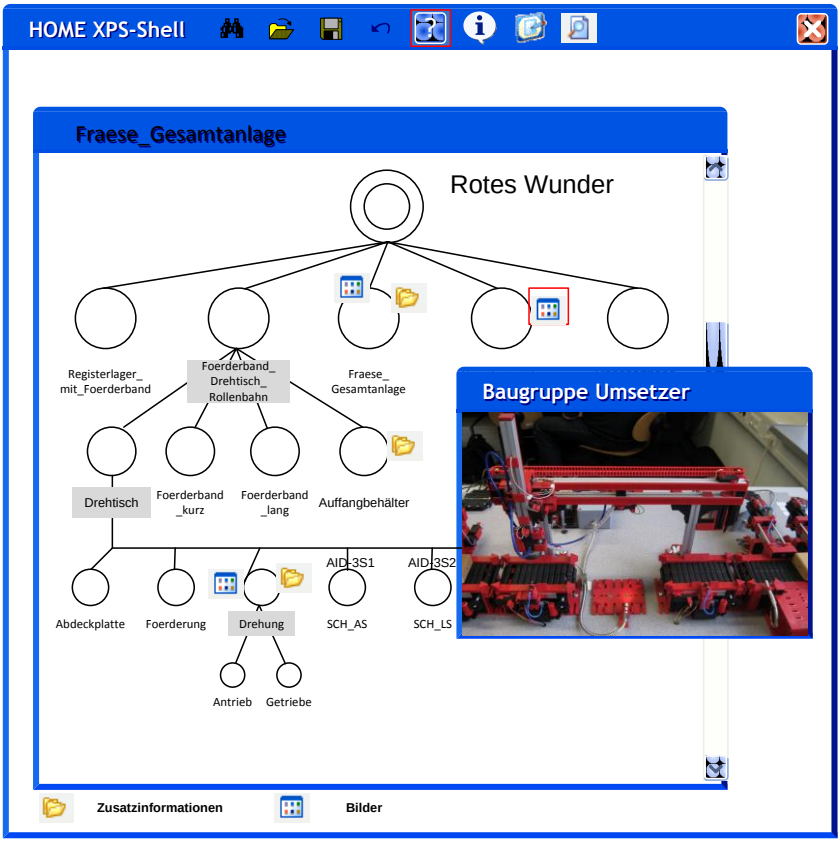


Abbildung 4.16: *Handbuch-Charakter eines XPS (Bauteilebaum)*

4.3.5 Notizbuchfunktion

Das Notizbuch dient dazu, dass der Anwender des Systems Informationen abspeichern kann, die später etwa vom KE verwendet werden können, um die Wissensbasis zu verbessern.
Damit dieser Effekt erreicht werden kann, sollten neben einer zur Kommentierung notwendigen freien Texteingabe folgenden Abspeicherungsmöglichkeiten im Notizbuch gegeben sein:

- die Wissensbasis mit allen Belegungen,
- der Bildschirminhalts zum Zeitpunkt des Aufrufs der Notizen,
- die Reihenfolge der Eingaben und Belegungen (Protokoll-Funktion).

Zwischen Notiz und behandeltem Objekt muss eine Beziehung dargestellt werden können, d.h. dass der Benutzer die Notiz zum Beispiel im Objektnetz positionieren kann, zumindest eine textuell eindeutige Zuordnung erfolgt. Fehlt diese Zuordnung werden die Notizen vor allem in großen Wissensbasen nicht leicht verarbeitbar, weil diese vielleicht mehrdeutig verstanden werden können, vor allem weil Benutzer und KE nicht die gleiche Person sind.
Notizen können bei der Pflege von Wissensbasen sehr hilfreich sein, müssen aber beständig verarbeitet werden.

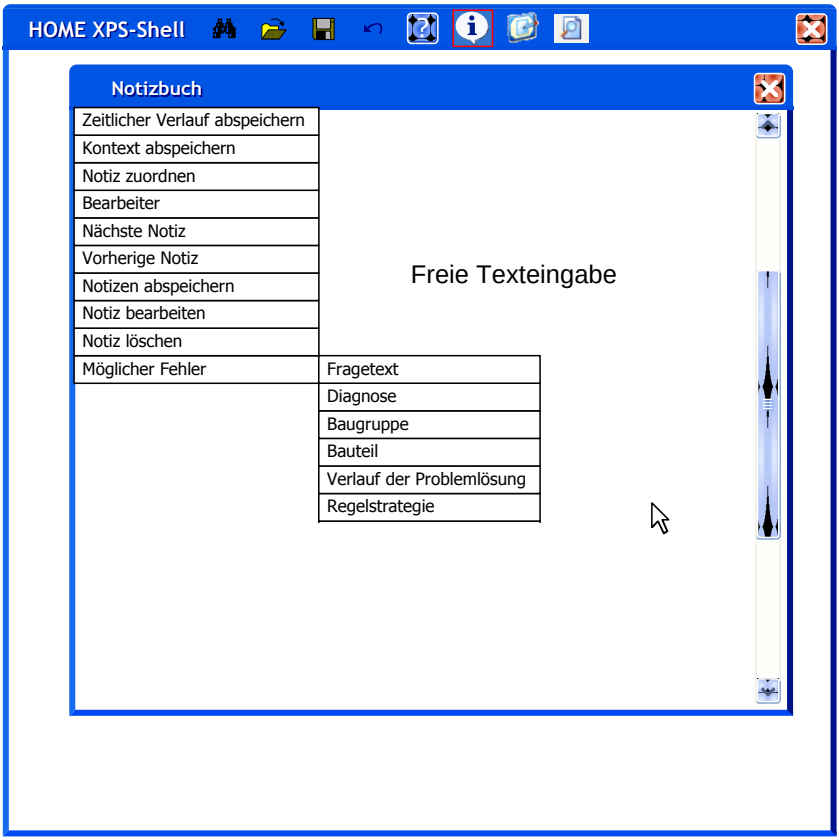


Abbildung 4.17: *Beispielhafter Aufbau eines Notizbuchs*

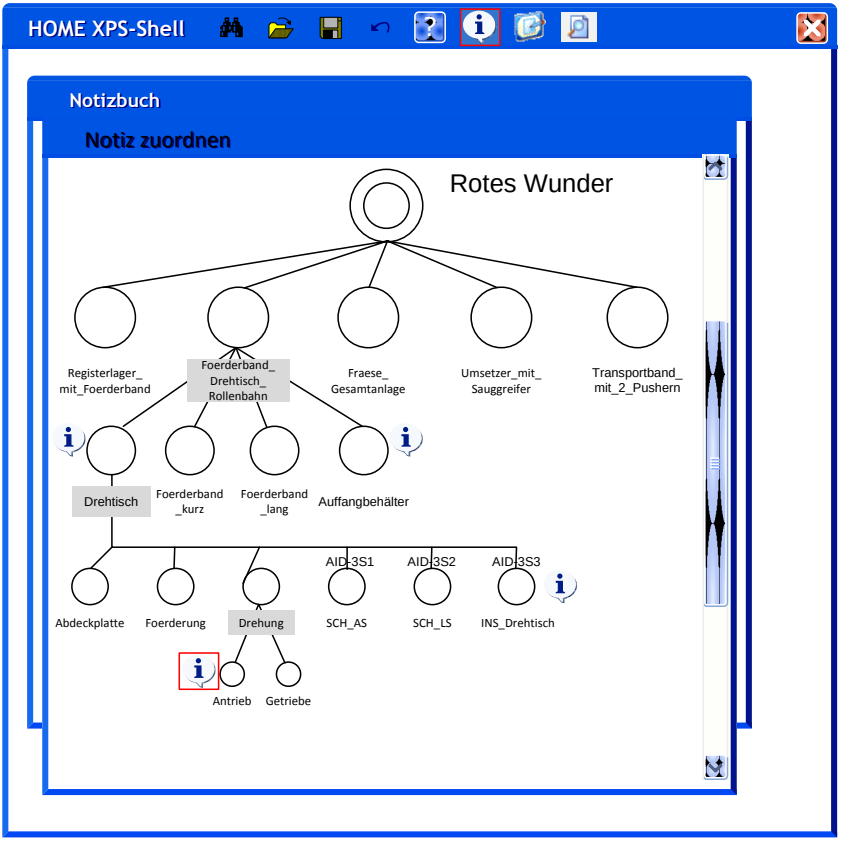


Abbildung 4.18: *Zuordnung einer Notiz zu einem Objekt*

4.3.6 Wissenspflegemodus

In diesem Modus muss es dem Benutzer möglich sein:

- Objekte zu definierten und zuzuordnen,
- Wissensbestandteile wie etwa Regeln syntaktisch korrekt zu verändern,
- freigestaltete Kommentare und/oder Medien an die verschiedene Objekte binden zu können; diese Funktion kann zum Beispiel zum Aufbau eines Handbuchs sinnvoll sein.

Dazu muss sichergestellt werden, dass der Benutzer eine fehlerfreie Eingabe von Wissensbestandteilen vornehmen kann. Dazu genügt ein einfacher Texteditor nicht mehr.
Zur Objekteingabe und Zuordnung ist hier ein Modus notwendig mit dem man Objekte in einem Netzwerk anordnen kann. Bei der Eingabe und Veränderung anderer Wissenselemente verwendet man in der Akquisitionskomponente meist sogenannte syntaxgesteuerte Editoren, die dem Benutzter die Eingabe verschiedener Schlüsselworte abnimmt, ihn aber auch auf einen bestimmten Verlauf der Eingabe festlegt. Beim Aufbau von Eingabemasken und Ausgabefeldern hat es sich bewährt diese an der gleichen Stelle anzuzeigen, wie sie während der Problemlösung erscheinen würden.
Man kann dabei davon ausgehen, dass durch eine Akquisitionskomponente nur kleinere Veränderungen einer Wissensbasis vorgenommen werden sollen, wie sie bei der Pflege von XPS nach deren Einführung vorgenommen werden. Eine große Wissensbasis so aufzubauen erscheint sehr zeitaufwendig, hier ist die Verwendung eines normalen Texteditors in Verbindung mit dem in Kapitel 4.2 beschriebenen Parser sinnvoller. Der Aufbau sollte in jedem Fall von einem versierten KE in Zusammenarbeit mit einem oder mehreren Experten vorgenommen werden. Ohne Kenntnisse der XPS-Technologie und der verschiedenen Abbildungsmöglichkeiten, halte ich einen normalen Benutzer hier für nicht geeignet, mag er auch Experte des Problembereichs sein.

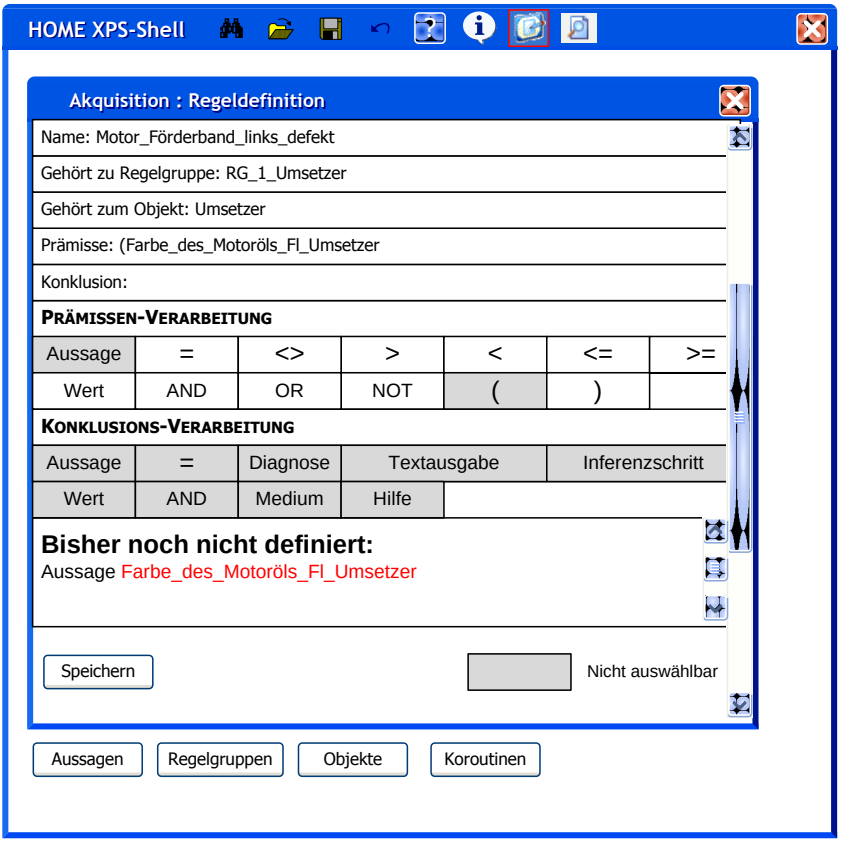


Abbildung 4.19: *Syntaxgesteuerter Editor (Beispiel Regeleingabe)*

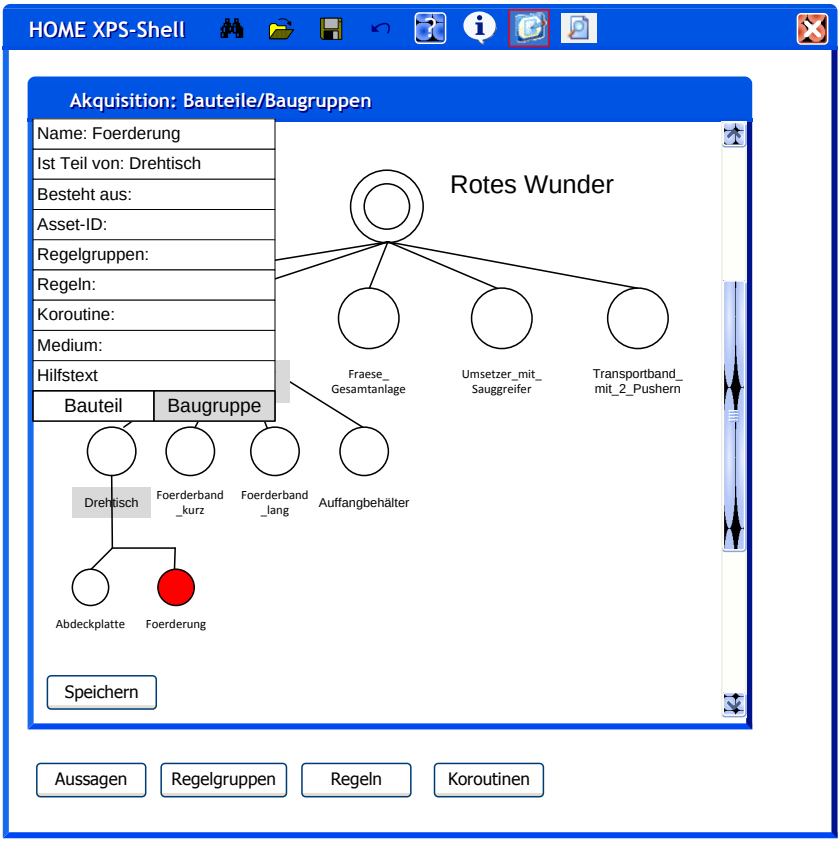


Abbildung 4.20: *Aufbau einer Objektstruktur*

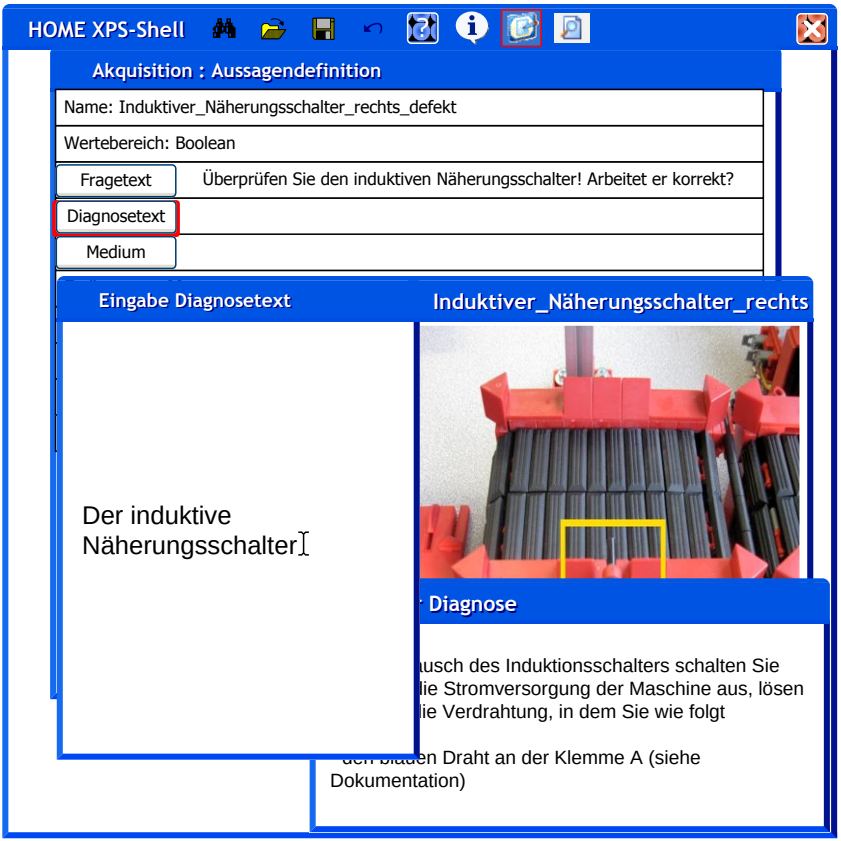


Abbildung 4.21: *Eingabe von Fragetexten und Kommentaren*

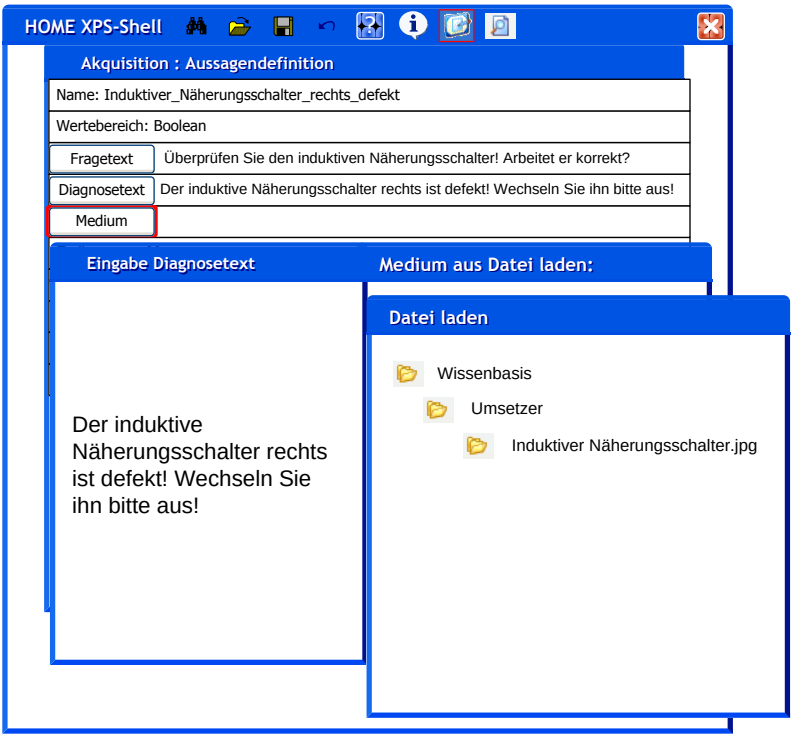


Abbildung 4.22: *Medienanbindung*

4.4 Die Erklärungskomponente

Die Erklärungskomponente wird dazu verwendet dem Benutzer:

- den Problemlösungsvorgang zu beschreiben,
- einen Kontext zu beschreiben (etwa die Bauteile zu beschreiben, die zur gleichen Baugruppe gehören) oder
- ihm einen Hinweis zu geben was von ihm in der Situation erwartet wird.

Die Ausgaben auf der Oberfläche wurden bereits im Kapitel 4.3 beschrieben. In diesem Kapitel möchte ich kurz auf die interne Struktur der Komponente eingehen, die im System realisiert sein muss, um überhaupt etwas erklären zu können.
Zur Abbildung des Problemlösungsvorgangs müssen folgende Bestandteile implementiert werden:

- eine LIFO-Liste, in der die Schritte vermerkt werden, die bisher während des Problemlösungs-Vorgangs durchgeführt wurden. Diese Liste wurde zu Beginn der KI mit dem Begriff Tracer bezeichnet;
- eine Struktur, die den Kontext in dem sich das System befindet speichert.

Die LIFO-Liste bildet dabei einen Stack, den der Benutzer nach und nach abgehen kann, und in dem ihm geschildert wird wie er in den momentanen Systemzustand kam.
Die Liste könnte etwa folgendermaßen aufgebaut sein:

- Schritt-Nummer,
- Wissenselement, z. B. Regel mit Name. Über diesen Namen könnet dann auf die Literale oder auf den Zusammenhang mit anderen Elementen verwiesen werden. Dabei stellen die entsprechenden Namen die Links zu den entsprechenden Elementen dar.

Der Kontext stellt den Zusammenhang des Elements mit anderen Elementen dar. So gehört eine Regel zum Bespiel zu einer Regelgruppe, die wiederum zu einem bestimmten Bauteil, dass wiederum zu einer Baugruppe gehört. Für den Nutzer kann es nun interessant sein wie das Umfeld des gerade betrachteten Elements aussieht. Zur Abspeicherung könnte man hier einfache Strukturen verwenden, die man bei Kontext-Wechsel in einer entsprechenden Liste abspeichert und dem Benutzer dadurch ermöglicht nicht nur die zeitliche

Veränderung zu betrachten, sondern auch den entsprechenden Kontext-Wechsel während des Problemlösungsvorgangs.
Um dem Benutzer entsprechende Hinweise zu geben, was in dieser Situation von ihm erwartet wird, muss man zwischen Standard-Erwartungen etwas „Was soll/kann ich jetzt tun“ und echten Hilfestellungen, die sich auf ein ganz bestimmtes Wissenselement bezieht „Wie kann ich das tun“ unterscheiden. „Was soll ich jetzt tun?“ analysiert den Zustand des Systems in einer bestimmten Situation, also etwa dem Benutzer wird ein Fragefenster ausgestellt und das System erwartet von ihm, dass er jetzt eine entsprechende Frage beantwortet. Hier wird also ein standardmäßiges Verhalten des Benutzers erwartet, also:

- Fragebeantwortung,
- Weiter bei der Ausgabe nach einer Problemlösung,
- Benutzung verschiedener Komponenten, also wie wird die Notizbuchfunktion benutzt,
- Verhalten, wenn das System auf einen Ablauffehler trifft, etwa wenn es zu einem Fehler innerhalb der Wissensbasis kommt
- Wenn kein Fehler gefunden werden konnte, etc.

Wenn der Nutzer allerdings wissen will, wie er etwas tun soll, müssen detaillierte Hilfen gegeben werden, also zum Beispiel wie ein Bauteil arbeitet, wann es zuletzt ausgetauscht wurde, wie es ausgebaut oder ausgetauscht werden kann… Dabei ist es auch sinnvoll entsprechende spezialisierte Mitarbeiter zu benennen.
Hilfstexte können dabei an alle Objekte einer Wissensbasis gebunden werden und können auch im Handbuch-Modus genutzt werden.

4.4.1 ein kurzer Blick in die Geschichte

Ein Ausgangspunkt der heutigen Erklärungskomponente in der KI ist das System SHRDLU, welches von Terry Winograd 1968-70 am M.I.T. entwickelt wurde /Stanford 10//Beierle 08/. Das im Deutschen auch als Block-Welt Programm bezeichnete, natürlichsprachliche System bildet eine Mikro-Welt ab, in der verschiedene einfache Objekte manipuliert werden konnten (siehe Abbildung 4.23).

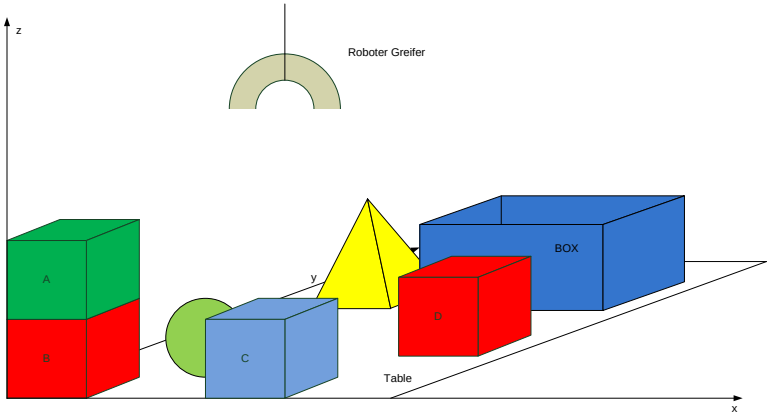


Abbildung 4.23: *Blockwelt mit einem typischen Zustand /Hartmann 84/*

Die Blockwelt besteht dabei aus einem Tisch, verschiedenen Objekten etwa Blöcken, einer Kugel, einer Pyramide und einer Schachtel, die auf einem Tisch stehen und durch einen Roboterarm manipuliert werden können. Den Objekten werden dabei verschiedene Eigenschaften der „primitiven Physik" wie "kann auf einem anderen stehen" , "kann als Stütze eines anderen Objekts dienen", "kann bewegt werden" etc. zugeordnet./Hartmann 84, Hartmann 01/ Das in Lisp auf einer DEC-Anlage entwickelte System konnte dabei seinen Weltzustand in einfachen Worten erklären also etwa „Block-A steht auf Block-B" „Block-A ist (2 2 2) Größeneinheiten groß" oder „Block-A ist grün". Zur Abbildung der zeitlichen Reihenfolge der Weltveränderung wurde eine Liste verwendet, die man als Trace bezeichnete und welche die Prozedur-Namen sowie die entsprechenden Parameter-Werte enthielt, also hatte der Trace zum Beispiel folgenden Inhalt:

```
((PUT-ON ´BLOCK-A ´BLOCK-B)(PUT-AT ´BLOCK-B ´(1 1 0)…).
```

Abschließend sei noch anzumerken, dass man durch die Beschäftigung mit SHRDLU zum ersten Mal auf ein Problem der Mikrowelten stieß. Fügt man nämlich in diese Welt ein Objekt nach dem anderen ein, so wird der Aufwand, die entsprechenden Objekte zu verwalten, ab einem bestimmten weiteren Objekt exponentiell steigen, weil ja der Rahmen der Welt nicht zu vergrößern ist. Der Tisch bleibt gleich groß.

4.5 Die Akquisitionskomponente

Da sich schon sehr bald in der Geschichte der Expertensysteme der Aufbau der Wissensbasis als eines der schwierigsten Probleme herausstellte, versuchten die Forscher Komponenten zu entwickeln mit deren Hilfe es dem Anwender selbst möglich sein sollte Wissen einzugeben und Fehler in den Wissensbasen zu beheben. Schon beim XPS-Großvater MYCIN wurde eine Komponente namens Teiresias /Hess 02//Spreckelsen 08//Steinmüller 05/ eingeführt, die diesen Wissenserwerb möglich machen sollte.

Soll eine Wissenserwerbskomponente einen Laien wirklich in die Lage versetzten komplexe Wissensbasen aufzubauen, so werden enorme Anforderungen an sie gestellt, weil der Laie keinerlei Kenntnisse über Wissensmodellierung und Abbildung besitzt. Die Komponente muss ihn also ständig führen und kontrollieren. Im Verlauf meiner industriellen Tätigkeit bei der Inpro haben wir uns auch mit der Realisierung einer solchen Komponente versucht /Dröll 89//Hartmann 85/. Diese wurde aber bald schon so komplex, groß und schwer zu handhaben, dass nach mehreren Jahren die Entwicklungsarbeit eingestellt wurde. Der Sinn der Komponente konnte den Mitarbeitern vor Ort niemals vermittelt werden.

Um einem Laien wirklich helfen zu können, ohne allerdings einen KE entbehrlich zu machen, sollte die Komponente:

- den syntaktisch korrekten Aufbau von Wissensbasen erlauben,
- den Benutzer beim Aufbau von übersichtlich gestalteten Objektnetzen unterstützen,
- Redundanzen und Inkorrektheiten in der Wissensbasis finden, die nicht syntaktisch verursacht werden.

Um darzustellen auf welche Probleme, man bei der Bewältigung dieser Ansprüche trifft, möchte ich hier zu den einzelnen Punkten einige Anmerkungen machen und Fragen aufwerfen, die man beim Aufbau einer Wissensbasis berücksichtigen sollte.

a) syntaktisch korrekten Aufbau von Wissensbasen

Beim Aufbau einer Wissensbasis müssen verschiedenen Wissenselemente wie Regeln, Literale etc. aufgebaut werden, die alle in einem Zusammenhang stehen. Baut man zum Beispiel eine Regel auf, verwendet man in der Prämisse Literale, die aus Bestandteilen bestehen, die noch nicht definiert sind. Dabei kann es dann vorkommen, dass die entsprechenden Aussagen vergessen werden, was vom System bemerkt werden muss. Eine syntaktisch fehlerhaftete Wissensbasis könnte möglicherweise zum Absturz des Systems führen. Das System muss also beim Laden den Fehler bemerken und den Nutzer dazu bringen alle nichtaufgelösten Referenzen zu bearbeiten.
Dazu müssen beim Aufbau der Elemente die Namen immer entsprechend vermerkt werden und ständig untersucht werden, ob vielleicht ein entsprechender Name schon vorhanden ist (zwei verschiedene Aussagen mit gleichem Namen dürfen nicht vorkommen). Der noch nicht definierten Namen muss dem Benutzer angezeigt werden. Wann aber stellt man diesen Namen aus. Nachdem die gesamte Regel aufgebaut ist oder direkt nachdem der Nutzer den Namen eingegeben hat?
Vielleicht ist die Regel einer anderen Struktur zugeordnet, etwa einer Regelgruppe. Wann sollte diese definiert werden, nachdem die Regel definiert wurde? Vielleicht gibt es dann aber auch noch keine weiteren Regeln in dieser Gruppe. Sollte man Regelgruppen mit einer Regel zulassen oder anzeigen?
Viele der Probleme werden sich im Laufe des Aufbaus der Wissensbasis auflösen. Dies kann aber sehr viel Zeit in Anspruch nehmen. Die Verwendung von syntaxgesteuerten Editoren kann bei den angesprochenen Problemen etwas helfen. Das Problem ergibt sich vor allem durch die Masse der einzugebenden Elemente.

b) übersichtlich gestalteten Objektnetzen

In den Wissensbasen hybrider Systeme stellen Objekte das Skelett dar, an dem die anderen Wissensbausteine angeordnet werden. Nur welche der Objekte stellt man dar? Die Expertensysteme, die ich in der Industrie im Bereich der technischen Diagnostik entwickelte, behandelten mehrere Tausend Bauteile. Wir verwendeten nur die Bauteile, die im Fehlerfall ausgetauscht werden (als Austauschtiefe bezeichnet).

Um diese Austauschtiefe zu ermitteln, sind umfangreiche Untersuchungen der Anlage vorab zu tätigen. Die verwendeten Stücklisten der Anlage sind möglicherweise nicht mehr aktuell. Welche Bauteile kann man zu Baugruppen zusammenfassen, welche Baugruppen wie in Bauteile aufteilen? Kommen einzelne Baugruppen eventuell leicht verändert in der Anlage mehrfach vor? Gibt es ganz spezielle Bauteile, die man auch besonders detailliert darstellen muss? Die Bauteile sollen zudem auch vielleicht noch im Handbuchmodus verwendet werden und müssen für diesen Fall gesondert behandelt werden.
An welche Bauteile bindet man welche anderen Wissenselemente, wo werden z.B. Regeln angeordnet? Mit welchen Bauteilen beginnt man die Problemlösung? Hilfsmöglichkeiten können hier eventuell graphische Editoren darstellen, mit deren Hilfe sich Objektnetze erstellen lassen /Parikh 89/.

c) Redundanzen und Inkorrektheiten in der Wissensbasis

Die einzelnen Wissensbestandteile werden in einer Wissensbasis über Ihren Namen angesprochen. Dabei kann es aber, vor allem bei großen Wissensbasen, vorkommen, dass unterschiedliche Wissenselemente den gleichen Namen besitzen oder die unterschiedlich bezeichneten Wissenselemente in Wirklichkeit das Gleiche meinen. Vor allem der zweite Fehler ist nicht leicht zu finden.
In der Wissensbasis kommen Elemente in Texten falsch bezeichnet werden. Dazu müssen alle Texte auch inhaltlich überprüft werden. Zumal jede Firma spezielle, nur dort verwendete Bezeichnungen besitzt ist dies meist sehr schwierig und wird von einem firmenfremden KE nur schwer bewerkstelligt werden können.
Die angesprochenen Probleme stellen nur einige wenige Beispiele dar. Jeder KE kann sie wahrscheinlich um mehrere Beispiele ergänzen. Die Forderung die Systeme selbst lernen zu lassen drängte sich daher schnell auf. Hier ergeben sich aber andere Probleme, die zum Teil im Gebiet des Maschinellen Lernens (siehe hierzu zum Beispiel /Michalski 83/) schon lange untersucht werden. Etwa:

- Welches Wissen ist wichtig und welches überfrachtet die Wissensbasis?
- Wie soll das aufgenommene Wissen eingebaut werden, es gibt wahrscheinlich mehrere Wissenselemente, die dies ermöglichen?
- Wie viel Hintergrundwissen soll aufgenommen werden?

- Wie werden hier Inkorrektheiten und Redundanzen gefunden?
- …

Alle angesprochenen Probleme lassen mich vermuten, dass egal welchen Aufwand man auch in Akquisitionskomponenten stecken mag, man auf eine durch einen versierten KE durchgeführte Wissensakquisition mit zahlreichen Untersuchungen, Interviews und einer Systemmodellierung nicht verzichten kann. Der Aufbau einer ernsthaft verwendbaren Wissensbasis dauert selbst dann noch bis zu einem Jahr.
Wissenserwerbskomponenten können von einem Laien auch heute noch nur zur Erstellung kleiner Wissensbasen und zur Korrektur bereist erstellter Wissensbasen verwendet werden. (siehe hierzu Kapitel 6)

4.6 Die Problemlösungskomponente

Die eigentliche Aufgabe eines XPS ist es Probleme innerhalb eines bestimmten Bereiches zu lösen. Aller anderen bisher behandelten Komponenten zeigten dem Benutzer die verschiedenen Lösungen an oder erklärten diese, die Problemlösungskomponente stellt hingegen das Herz der Verarbeitung dar. Sie greift auf die Wissensbasis zu, um Probleme die vom Benutzer gestellt werden zu bearbeiten, und stellt während dieses Vorgangs Daten in einer Form bereit, dass dem Benutzer ihr Vorgehen erklärt werden kann. Dabei verarbeite sie Daten des Benutzers oder anderer Datenquellen etwa einer Schnittstelle zu einer Steuerung oder einer Datenbank.

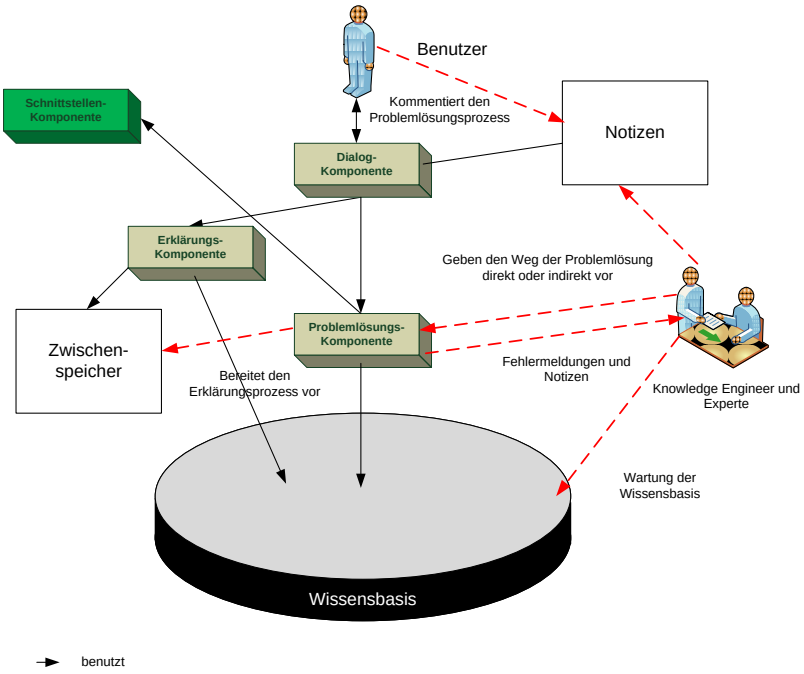


Abbildung 4.24: *an der Problemverarbeitung beteiligten Komponenten mit Datenquellen*

Die Problemlösungskomponente enthält dabei verschiedene Strategien zur Verarbeitung des in der Wissensbasis dargestellten Wissens, welches im Wesentlich durch seine Struktur die Verarbeitung bestimmt. Die Verarbeitung ist dabei wesentlich von der Eingabe des Nutzers und der einbezogenen Datenquellen abhängig, weshalb die Problemlösung viele verschiedene Variationen zulässt. Dies stellt die Entwickler vor allem beim Testen einer größeren Wissensbasis vor Probleme.
Die Arbeit einer Problemlösungskomponente ist von der Repräsentationsform des Wissens abhängig. Da es sehr viele verschiedene Repräsentationsformen gibt, würde der Versuch hier alle zu behandeln den Rahmen dieses Buches sprengen. Ich habe mich deshalb hier auf die meistverwendeten Repräsentationsformen Regelverarbeitung, Objektverarbeitung (Frames) und der Kombination beider beschränkt.

4.6.1 Generelle Regelverarbeitung

Die hier verwendete Regeln lehnen sich an die Logik an, ohne mit der Verarbeitung dort vollständig übereinzustimmen (siehe hierzu auch /Hartmann 09/).
Eine Regel zerfällt dabei grundsätzlich in eine Prämisse und eine Konklusion. Die Prämisse stellt dabei die Voraussetzung dar, aus der bei einem logischen Schluss auf ein Ergebnis, die Konklusion, geschlossen wird / Brockhaus 10/. Gewöhnlich wird diese Verarbeitung als WENN-DANN-Struktur bezeichnet und ähnelt einer IF-THEN-Struktur einer Programmiersprache. Im Gegensatz zu herkömmlichen IF-THEN-Strukturen ist die Reihenfolge der Verarbeitung hier aber nicht festgelegt, sondern ergibt sich erst innerhalb des Problemlösungsvorgangs /Knemeyer 94/.

WENN	(Literal ODER Literal) UND NICHT Literal	DANN	Literal UND Literal
	PRÄMISSE		***KONKLUSION***

Prämisse und Konklusion setzen sich dabei aus mit logischen Operatoren verknüpften Literalen zusammen. Da mit den Operationen Konjunktion (UND), Disjunktion (ODER) und Negation (NICHT) alle logischen Strukturen abgebildet werden können (vergleiche hierzu /Böhme 81/ und /Hartmann 09/) müssen nur diese abgebildet werden. Bei der Darstellung der

logischen Operationen sind dabei auch andere Codeworte oder Symbole gebräuchlich, wie etwa AND, Λ, OR, V, NOT, ¬ etc..
Literale können dabei Aussagen, Zuweisungen oder Vergleiche sein (siehe hierzu Kapitel 4.2). Die Problemlösungskomponente muss nun überprüfen ob das Literal „wahr" ist. Dabei muss nach dem in der Wissensbasis festgelegten Bestandteilen der entsprechenden Elemente vorgegangen werden. So wird einer Aussage einen Fragetext, einen Wertebereich und eventuell einen Diagnosetext als Bestandteile zugewiesen.
Innerhalb der Problemlösungskomponente wird beim Auftreten der entsprechenden Aussage in einem Literal der Prämisse dem Benutzer der Fragetext ausgestellt und von diesem eine entsprechend dem Wertebereich eingeschränkte Werteauswahl gefordert.

Beispiel:
In der Wissensbasis ist die Regel *Motor-Öl-verschmutzt* aufgeführt, zu der im Verlaufe der Problemlösung gekommen wird.

Regel *Motor-Öl-verschmutzt*
WENN Farbe_des_Öls = „schwarz" DANN DIAGNOSE Öl_auswechseln

Der Inferenzer greift auf diese Regel zu und verarbeitet zunächst die Prämisse (WENN-Teil). Er stellt fest, dass die Prämisse aus einem Vergleich einer Aussage mit einem Wert besteht. Er überprüft zunächst, ob die Aussage bereits einen Wert hat. Ist dies nicht der Fall, stellt er den Fragetext aus und erwartet eine Eingabe des Benutzers, wobei die möglichen Werte der Aussagen angegeben werden. Ist der Wert bereits belegt wird dieser Wert bei der weiteren Verarbeitung verwendet.

Die Aussage *Farbe_des_Öls* wird in der Wissensbasis jetzt wie folgt definiert:

Aussage *Farbe_des_Öls*
Fragetext: „Schauen Sie sich die Farbe des Öls am Motor an. Welche Farbe hat es?"
Wertebereich: Liste [honigfarben, verschmutzt, schwarz]
WERT: NIL
N aus M: false
Diagnosetext:
Medien: Bild „Öl_testen.gif"

Da der Bestandteil „Wert“ noch nicht belegt ist, wird dem Benutzer der entsprechende Fragetext und das entsprechende Bild ausgestellt und die Werte *honigfarben*, *verschmutzt* und *schwarz* zur Auswahl ausgestellt. Der Bestandteil N aus M sagt hier aus, dass nur ein Wert auszuwählen ist. Wäre der Bestandteil mit true belegt könnte der Benutzer mehr als einen Wert auswählen, was bei dieser Aussage nicht sinnvoll wäre.

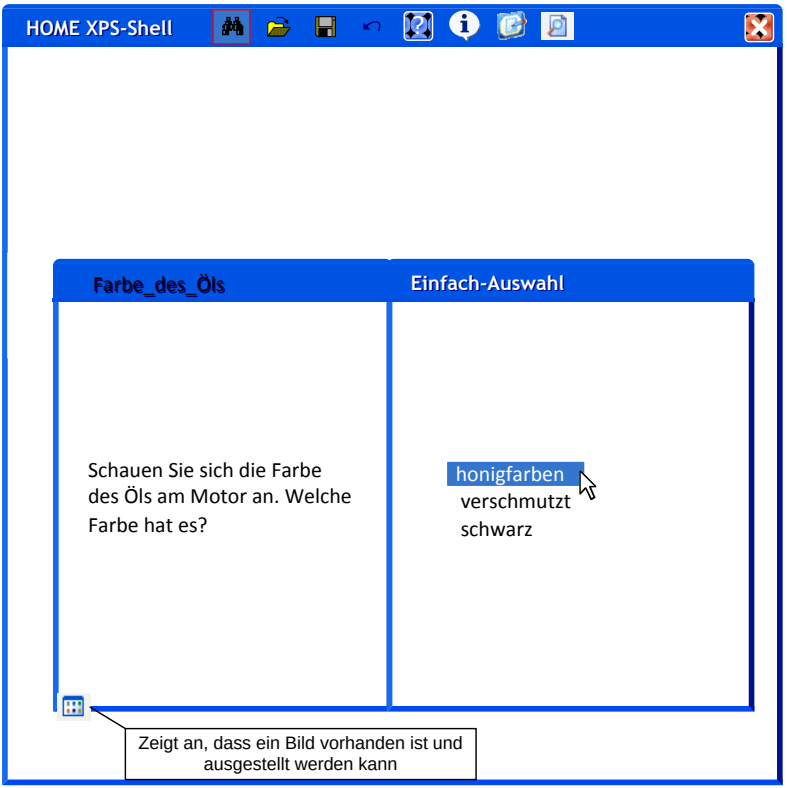

HOME XPS-Shell
Farbe_des_Öls
Einfach-Auswahl
Schauen Sie sich die Farbe des Öls am Motor an. Welche Farbe hat es?
honigfarben
verschmutzt
schwarz
Zeigt an, dass ein Bild vorhanden ist und ausgestellt werden kann

Abbildung 4.25: *Ausgabe des Beispiels*

In der externen Wissensbasis wird normalerweise keine Vorbelegung einer Aussage vorgenommen, so dass der Wert beim Laden der Wissensbasis unbelegt ist. Eine Aussage kann aber mehrfach in einer Wissensbasis vorkommen, so dass die Aussage bereits einen Wert haben kann. Um dies aber innerhalb der Problemlösung überhaupt feststellen zu können, muss es einen internen Bestandteil der Aussage geben, der anzeigt, dass diese bereits belegt wurde. Dieser Bestandteil wird meist als TRACED bezeichnet, und zu Beginn der Problemlösung oder beim Rücksetzten der Wissensbasis auf false gesetzt. Der Inferenzer überprüft diesen Bestandteil und verwendet, sollte TRACED gesetzt sein, den Inhalt des Bestandteils WERT.
Nachdem der Benutzer eine Auswahl getätigt hat, wird diese dem Bestandteil WERT der Aussage zugeordnet und der Bestandteil TRACED vom Inferenzer auf true gesetzt.
In der Prämisse der Regel *Motor-Öl-verschmutzt* wird nun der Vergleich des tatsächlichen Wertes mit dem dort angegebene Wert vorgenommen. Hat das Literal Farbe_des_Öls also tatsächlich den Wert *schwarz,* dann wird der Vergleich vom Inferenzer als wahr erkannt und da keine weiteren Bestandteile in der Prämisse vorhanden sind, wird diese insgesamt als wahr angenommen. Man sagt dann, dass die Regel „feuert". Die Konklusion wird ausgeführt, der WERT des Literals *Öl_auswechseln* sowie TRACED mit true belegt.
Beim Vergleich zweier Aussagen in der Form Aussage1 = Aussage2 wird der Inferenzer die beiden WERTE der Aussagen eventuell nach deren Belegung durch den Nutzer vergleichen. Wird eine Aussage ohne Vergleichen und Wert angegeben, so ist der Wertebereich der Aussage einfach Boole, so dass damit nur eine Abkürzung für Aussage = true verstanden werden kann. In jedem Fall überprüft der Inferenzer den Bestandteil TRACED und belegt diesen, so dass die Aussage im späteren Verlauf der Problemlösung nicht erneut ausgestellt werden muss.
Sieht die Wissensrepräsentation auch die Verwendung von Funktionen vor, so muss in der Prämisse der von diesen zurückgelieferten Werten betrachtet werden. Auch dabei kann ein Wert oder ein true bzw. false zurückgeliefert werden.
Die Prämisse einer Regel kann sich aus verknüpften Literalen zusammensetzen. Die Verarbeitung einer solchen verknüpften Prämisse erfolgt dabei nach den aus der Logik bekannten Verarbeitungsvorschriften der Konklusion, Disjunktion und Negation.
Eine Konklusion kann aus einer Verknüpfung von Literalen bestehen, wobei diese Verknüpfungen aber nicht den logischen Verknüpfungen entsprechen sondern einfach als Sequenz zu verstehen ist, die nacheinander auszuführen

sind. Eine Konklusion kann dabei aber auch andere Bestandteile haben etwa TEXTAUSGABE, DIAGNOSEAUSGABE oder MEDIENAUSGABE. Bei der TEXTAUSGABE kann dabei ein freier Text ausgegeben werden, ohne dass eine entsprechende Aussage angefasst werden muss. Bei der DIAGNOSEAUSGABE *Aussagex* wird der Diagnosetext der Aussage *Aussagex* ausgegeben, bei MEDIENAUSGABE *Aussagey* das entsprechende Bild, Video etc. der *Aussagey*.

HOME XPS-Shell
Textausgabe Motor_defekt
Motor_Umsetzer_rechts
Der Motor ist durch verschmutztes Öl beschädigt, wechseln Sie ihn bitte aus!

Abbildung 4.26: *Ausgabe einer Diagnose*

Der Problemlöser nimmt vor der Verarbeitung der Regeln keine Überprüfung auf die syntaktische Korrektheit vor, so dass es zum Beispiel zu Fehlern

kommen kann, falls Aussagen überhaupt nicht vorhanden sind oder Bestandteile wie etwa der Fragtexte fehlen.
Die ersten rein regelbasierte Systeme hatten eine einzige Regelbasis aus der die verschieden zu verarbeitenden Regeln ausgewählt werden mussten. In einem sogenannten Match-Execute Zyklus (siehe Abbildung 4.27) wurden verschiedene Regeln ausgewählt, deren Verarbeitung wieder zu neuen zu betrachten Regeln führte usw. Dies war auch notwendig, weil man sehr große Regelbasen zu verarbeiten hatte und nicht alle tausenden Regeln durcharbeiten wollte. Bei modernen XPS wird diese Verarbeitung noch in den speziellen Verarbeitungsarten verwendet, wo sie aber nur auf eine eingeschränkte Regelbasis angewendet wird.

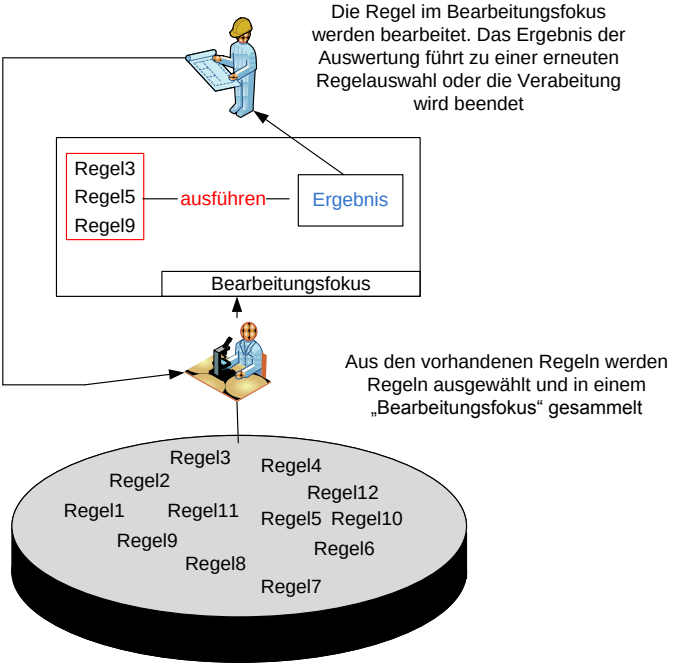


Abbildung 4.27: *Match Execute Zyklus /Knemeyer 94/*

Moderne Wissensbasen, die heute vorwiegend hybride Repräsentationen verwenden, teilen die Regeln hingegen direkt in unterschiedliche Regelgruppen, die den verschiedenen Objekten zugeordnet werden. So können zum Beispiel im Bereich der Diagnostik die Regeln zu einer bestimmten Hypothese zu einer entsprechenden Regelgruppe zusammengefasst werden.

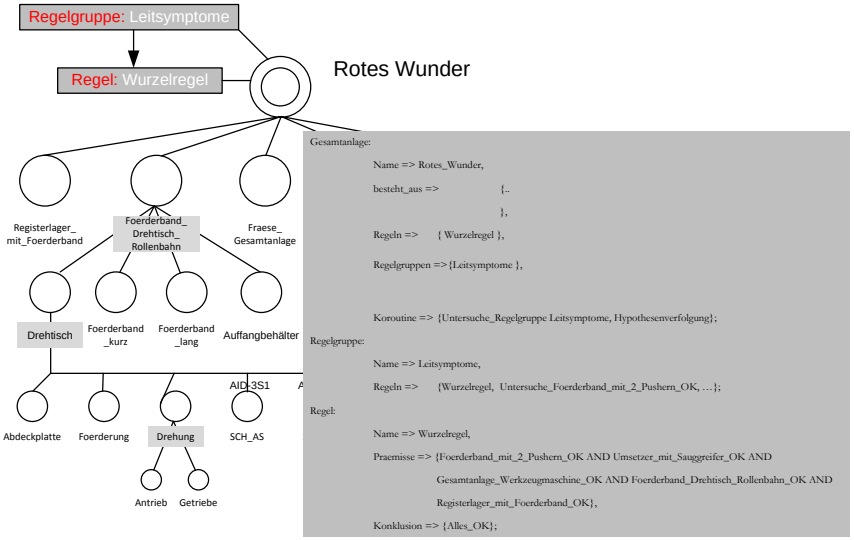


Abbildung 4.28: *Zuordnung der Regelgruppen-Objekten (hybride Systeme)*

Innerhalb der Regelgruppen, die nur wenige Regeln zu umfassen brauchen, können dann die folgenden Strategien verwendet werden:

-sequentielle Verarbeitung
Alle Regeln der Regelgruppe werden durchgegangen. In großen Wissensbasen früherer XPS hätte dies sehr lange gedauert und vielleicht dennoch zu keinem Ergebnis geführt;

- gewichtete Verarbeitung (Priorität)
Dabei könnten die Regel zum Beispiel unabhängig danach wie der Wissensingenieur sie in die Wissensbasis geschrieben hat, so sortiert werden,

dass etwa die kürzesten Prämissen am Anfang stehen oder die Sortierung so erfolgt, dass die Regeln mit den meisten bereits belegten Aussagen vorne in der Regelgruppe stehen. In beiden Fällen wird versucht dem Benutzer Eingaben zu ersparen. Auch eine Priorisieren verschiedener Regeln ist hier möglich (siehe hier auch Gewichtungsfaktoren);

- die im folgenden behandelten speziellen Verarbeitungsarten Vorwärts- und Rückwärtsverkettet und Goalbasiert. (sieh hierzu auch /Groschwitz 99/)

4.6.2 Spezielle Regelverarbeitung

In dem folgenden Beispiel beziehe ich mich auf die Diagnostik einer Transferstraße, wie ich sie in der Praxis kennen gelernt habe. Dabei soll eine Bohrstation diagnostiziert werden. Die Station bohrt ein Loch ihn ein Werkstück, dass ihr über ein Transportband zugeführt wird und besteht aus:

- einer Schlitteneinheit, welche aus
    - Schlittenbett und
    - Schlitten mit
        - Motor
        - Bohrer

  besteht
- einem Endschalter
- einem Anfangspostionsschalter .

Die Station wird zentral mit Kühlwasser und Strom versorgt. (Abbildung 4.29)

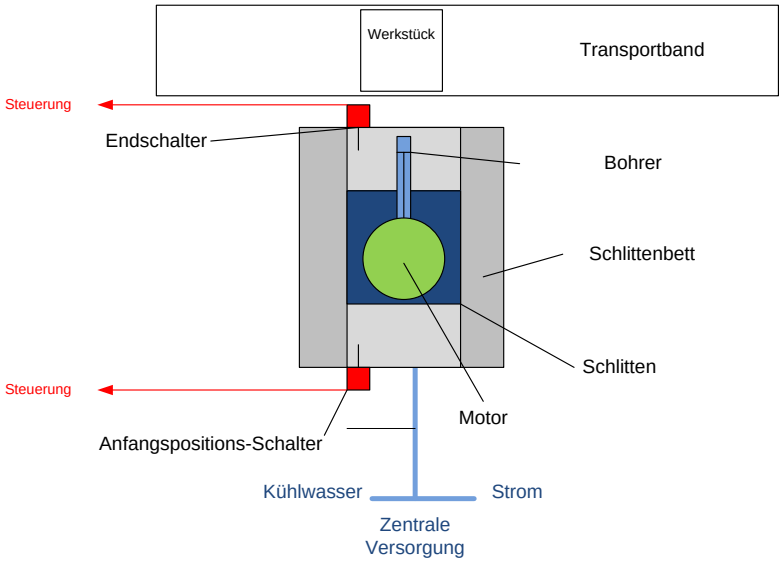


Abbildung 4.29: *Schematische Darstellung der Bohrstation*

a) Vorwärtsverkettung

Bei der Vorwärtsverkettung werden die Prämissen der Regeln einer Regelmenge betrachtet. Ausgehend von bereits belegten Aussagen werden die Regeln dahin überprüft, ob sie feuern können. Ist dies der der Fall werden die sich daraus ergebenden neuen Aussagen mit in die Verarbeitung genommen und die Regelmenge erneut bearbeitet bis keine Regel mehr feuern kann.

Der erste Schritt einer Vorwärtsverkettung ist es, eine geeignete Regelmenge zu finden. Im Beispiel sei angenommen, dass in einem Diagnose-Expertensystem geprüft werden soll ob eine Station einer Transferstraße, in der ein Loch in ein Werkstück gebohrt wird, korrekt arbeitet.

Der fehlerfreie Vorgang kann so beschrieben werden:

Eine Schlitteneinheit mit einem Schlitten, auf dem ein Motor installiert ist fährt aus der Ausgangsposition bis zur Endposition.

Da der Motor läuft und einen Bohrer antreibt wird dieser durch diesen Vorgang ein Loch bohren.

An der Endposition angekommen wird geprüft ob das Loch vorhanden ist und der Schlitten dann wieder zurückgefahren, bis er seine Endposition erreicht.

Wenn alles in Ordnung ist (OK_Signal) wird das Band mit den Werkstücken um eine Position weitergefahren und der Vorgang wiederholt sich.

In unserem Fall liegt nun ein Fehlerfall vor, der untersucht werden soll. Die Aussagen A, B und C können durch die Anfrage an eine Steuerung belegt werden, alle anderen Aussagen sollen erschlossen werden. Die Anfrage an die Steuerung erfolgt, wenn die entsprechende Prämisse verarbeitet wird. Abgebildet ist hier ein Großteil der Regeln, die eine korrekte Abfolge des Prozesses beschreiben. Es wird dabei davon ausgegangen, dass die zusammengesetzten Aussagen in der Regelmenge zuerst, die einfachen danach bearbeitet werden sollen. Ich habe das Beispiel an eine von /Kneymeyer 94/ vorgegebenes Beispiel der Vorwärtsverkettung ausgereichte und es auf den Bereich Diagnostik angepasst.

Beispiel Vorwärtsverkettung Fehler bei Bohrvorgang:

Gegeben die Aussagen mit folgender Bedeutung

Aussage	**Klartext**	**Belegt von**
A	Strom_an	Steuerung
B	Loch_gebohrt	Steuerung
C	Kühlwasser_vorhanden	Steuerung
D	Motor_läuft	Inferenz oder erfragt
E	Schlitten_in_Endposition_Signal	Inferenz oder erfragt

F	OK_Signal_geben	Inferenz oder Steuerung
G	Fehler_ausgabe	Inferenz

Regelname	Prämisse	Konklusion
R1	WENN A	DANN D
	WENN Strom_an	DANN Motor_läuft
R2	WENN NICHT F	DANN G
	WENNNICHT OK_Signal_geben	DANN Fehler_ausgeben
R3	WENN B UND D	DANN E
	WENN Loch_gebohrt UND Motor_läuft	DANN Schlitten_in_Endposition_Signal
R4	WENN E UND C DANN F	
	WENN Schlitten_in_Endposition_Signal UND Kühlwasser_vorhanden	DANN OK_Signal_geben

Die Regelmenge sei wie folgt geordnet:
R4
R3
R1
R2

Der Inferenzer bearbeitet die so definierte Regelgruppe. Er greift die Regeln nacheinander an, wobei er die einfachen Regeln zuerst überprüft. R4 und R3 sind zusammengesetzte Regeln, er führt deren Prämissen nicht aus. R1 ist die erste einfache Regel, A ist von der Steuerung abzugreifen, an dieser Stelle beginnt die Vorwärtsverkettung.

1. Durchgang durch die Regelmenge

   R1 feuert, womit die Aussage D belegt werden kann. Die Regelmenge wird erneut durchgearbeitet.

2. Durchgang durch die Regelmenge

R4 wird zwar angefasst, die Aussagen E in der Prämisse kann von der Steuerung aber nicht belegt werden und ist auch nicht belegt.
R3 aber kann feuern, weil D belegt ist und B durch eine Anfrage an die Steuerung belegt werden kann. Durch das feuern der Regel wird E belegt und kann im nächsten Durchgang verwendet werden.

3. Durchgang durch die Regelmenge

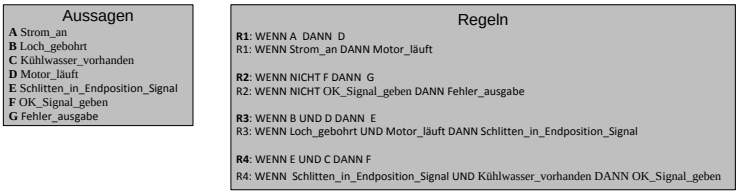
R4 kann nun angefasst werden. Ein Fehlerfall soll vorliegen und dieser wird jetzt auch gefunden, indem der Inferenzen die Steuerung abfragt und bemerkt, dass kein Kühlwasser vorhanden ist. Damit wird F nicht belegt.

4. Durchgang durch die Regelmenge

Die letzte Regel, die noch nicht angefasst wurde ist R2. Würde kein Fehlerfall vorliegen würde die Prämisse angegriffen und festgestellt, dass OK_Signal_geben mit true belegt wäre(im 3. Durchgang durch R4), die Regel würde also nicht feuern. Die Regelmenge wäre ganz verarbeite der Inferenzer würde „kleinlaut“ eingestehen müssen, dass er den Fehler nicht findet.
Da aber die Prämisse von R4 nicht erfüllt werden konnte wird die Prämisse von R2 angegangen und es wird festgestellt, dass OK_Signal_geben noch nicht belegt ist.

ACHTUNG: Wenn etwas nicht belegt ist, ist es nicht automatisch mit NICHT belegt!

Daher wird hier die Steuerung befragt(die Steuerung hat diese Aussage C ja nicht – wie vorgesehen - automatisch belegt) und dieser stellt fest das, das OK_Signal nicht gegeben wurde. Als Resultat der Steuerungsüberprüfung wird OK_Signal = false gesetzt und R2 feuert, als Fehlerausgabe könnte nun ausgegeben werden:
„Entweder ist der Schlitten nicht in die Endposition angekommen (Schlitten_Endposition Signal fehl) oder es fehlt das Kühlwasser!“
Erst eine weitere Überprüfung von R4 könnte den Fehler hier weiter einschränken.
Zur Korrekten Arbeit müssten auch die Zwischenschritte mit Fehlerausgaben versehen werden!



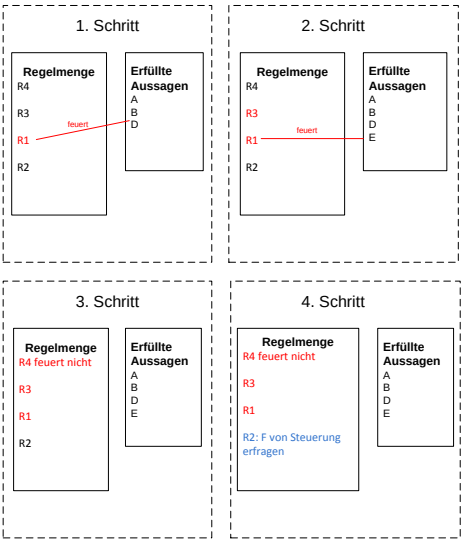


Abbildung 4.30: *kurze schematische Darstellung der Vorwärtsverkettung*

b) Rückwärtsverkettung/Goalbasierte Verarbeitung

Bei der Rückwärtsverkettung steht die Konklusion im Fokus. Bei der Verarbeitung einer Prämisse wird zunächst betrachtet ob das entsprechende Literal auch in der Konklusion einer anderen Regel steht, also von dieser Regel abgeleitet werden kann.

Wird eine einzelne Regel oder eine Aussage rückwärtsverkettet, wird bei der Verarbeitung der Prämisse zunächst versucht die entsprechenden Aussagen in der Konklusion anderer Regeln aufzufinden und deren Prämisse zu bearbeiten. Gelingt dies nicht wird meist der Nutzer direkt gefragt. Diese Art der

Verkettung wird auch in modernen Systemen noch verwendet, wenn zum Beispiel eine Beantwortung einer Frage den Benutzer überfordern würde. Fragt man den Benutzer zum Beispiel ob alle Leitungen OK sind, vergisst er vielleicht eine Leitung. Man bietet ihm also rückwärtsverkettet eine Regel a, in der er einmalig alle Leitungen überprüfen muss und belegt damit eine Aussage „alle_Leitungen_überpüft", die der Nutzer überhaupt nicht sieht. Im weiteren Verlauf der Problemlösung verwendet man ausschließlich diese Aussage. Dies hat den Vorteil, dass man die vielleicht längere Regel zur Überprüfung aller Leitungen nur einmal definieren muss, andere Regeln kürzer werden und die Wissensbasis überschaubarer wird.

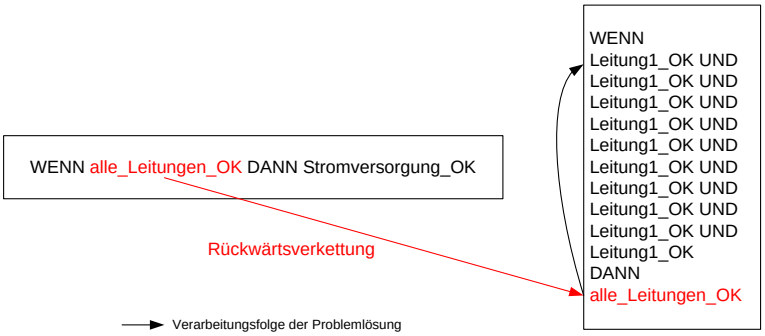


Abbildung 4.31: *Beispiel der Rückwärtsverkettung*

Die Goalbasierte Inferenz ist dabei eine konsequente Rückwärtsverkettung über die gesamte Regelmenge.
Das System startet mit einer Regel, welche das Goal in der Konklusion hat und dessen Prämisse, aus rückwärtsverketteten Aussagen besteht. Über diese Aussagen wird dann innerhalb der Verarbeitung auf die enstprechenden Regeln verwiesen. Das System endet, wenn das Goal belegt wurde (siehe Beispiel). Diese ganz konsequente Rückwärtsverkettung über eine ganze Regelmenge wird heute nicht mehr verwendet, weil es in den hybriden Systemen heute andere Modellierungsstrukturen gibt (siehe hierzu auch Kapitel 5 und 6).

Beispiel für die Goalbasierte Inferenz:

Aussage	**Klartext**	**Fragetext**	**Wertebereich**

A	Motor_defekt		
B	Endschalter_defekt		
C	Anfangspositions_schalter_defekt		
D	Kühlwasser_fehlt		
E	Strom_fehlt		
A_1	Überprüfe Motor	„Überprüfen Sie die Funktion des Motors! Läuft er korrekt:“	Boole
B_1	Endschalter_Kontrolle	„Wird das Signal von der Schlitteneinheit in Endposition von der Steuerung angezeigt! “	[„angezeigt“ „nicht angezeigt“]
C_1	Anfangskontrollschalter_OK	„Wird das Signal von der Schlitteneinheit in Anfangsposition von der Steuerung angezeigt! “	[„angezeigt“ „nicht angezeigt“]
D_1	Prüfung_Kühlwasser	„Kühlwasser vorhanden?“	Boole
E_1	Prüfung_Strom (rückwärtsverkettet)	„Strom vorhanden?“	Boole
E_2	Stromleitung überprüfen	„Überprüfen Sie die Stromleitungen 1 bis 3! Sind diese OK?“	Boole
Goal	zu belegende Startpunkt		

Die bei der Rückwärtsverkettung verwendeten Regeln:

Name	
R1	WENN NICHT A UND NICHT B UND NICHT C UND NICHT D DANN Textausgabe „Kein Fehler feststellbar“ UND GOAL=“kein Fehler“ UND Ausgabe Goal
R2	WENN A ODER B ODER C ODER D ODER E DANN TEXTAUSGABE „Fehler: „ UND Ausgabe GOAL

R3 WENN A_1 DANN NICHT A (Motor_defekt = false)
R3$_1$ WENN NICHT A_1 DANN A (Motor_defekt = true) UND GOAL ="Motor"
R4 WENN B_1 = „angezeigt" DANN NICHT B (Endschalter_defekt = false)
R4$_1$ WENN B_1 = „nicht angezeigt" DANN B (Endschalter_defekt = true) UND GOAL ="Endschalter defekt"
R5 WENN C_1 = „angezeigt" DANN NICHT C (Anfangspositions-Schalter_defekt = false)
R5$_1$ WENN C_1 = „nicht angezeigt" DANN C (Anfangspositions-Schalter_defekt = true) UND GOAL ="Anfangspositions-Schalter defekt"
R6 WENN D_1 DANN NICHT D (Kühlwasser_fehlt = false)
R6$_1$ WENN NICHT D_1 DANN D (Kühlwasser_fehlt = true) UND GOAL ="Kühlwasser fehlt"
R7 WENN E_1 DANN NICHT E (Strom_fehlt = false)
R7$_1$ WENN NICHT E_1 DANN E (Strom_fehlt = true) UND GOAL ="Stromversorgung defekt"
R8 WENN E2 DANN E_1 (Stromleitung_überprüfen = true)
R8$_1$ WENN NICHT E_2 DANN NICHT E1 (Stromleitung_überprüfen = false)

Verarbeitung:

Schritt 1: das System startet in dem es die Aussage Goal belegen möchte und nach Regeln sucht, die dies tun. Gesucht werden also Regeln mit der Aussage Goal in der Konklusion.
Gefunden werden
die Regel R1 und R2.

Schritt 2:
Die Regel R1 wird zunächst verarbeitet. Bei der Prämissenverarbeitung wird die Aussage A gefunden und rückwärtsverkettet, d.h. es wird in einer anderen Regel in der Konklusion gesucht. Wird A gefunden wird diese Regel als nächstes verarbeitet. R3 und $R3_1$ werden gefunden.
Schritt 3:
R3 wird verarbeitet. Die Aussage A_1 wird in einer anderen Regelkonklusion gesucht und nicht gefunden, also muss sie erfragt werden. Der entsprechende Fragetext wird ausgestellt der Benutzer hat dann die Möglichkeit mit true oder false zu antworten. Antwortet er mit true wird der Motor_defekt auf false gesetzt. Die Regel $R3_1$ feuert nicht, weil die beiden Prämissen sich gegenseitig ausschließen. Wird A_1 hingegen mit false belegt feuert R3 nicht, die Belegung wird dann über $R3_1$ vorgenommen und die Aussage Goal wird gesetzt. Da die

Aussage jetzt gesetzt ist, wird in die Regel R1 zurückgesprungen und die Prämissenverarbeitung fortgesetzt.

Schritt 4 bis n:
Solange kein Fehler gefunden wird die Prämisse von R1 abgearbeitet. Wir hingegen ein Fehler gefunden, was bedeutet das eine der Aussagen A bis E mit true belegt wurde, wird die Prämisse abgebrochen, weil ja alle Literale true ergeben müssen, damit die Regel feuern kann. In diesem Fall wird die Regel 2 sofort feuern, weil hier nur ein Literal true annehmen muss.
Die Verarbeitung endet wenn das Goal belegt wurde, was entweder durch die Regel R1 oder R2 erfolgt.

Die Rückwärtsverfolgung kann dabei über mehrere Stufen gehen, wie ich in den Regel R7, $R7_1$, R8 und $R8_1$ zeigen wollte, in denen die Aussage E über die Aussagen E_1 und E_2 mehrstufig belegt werden. Über die, den einzelnen Wissenselementen zugeordneten Strategie kann der KE festlegen ob eine Aussage rückwärtsverkettet werden soll oder der voreingestellten Strategie (meist Frage-Benutzer) gefolgt wird. Zum Einsatz der Strategien möchte ich auf Kapitel 4.6.5 verweisen.

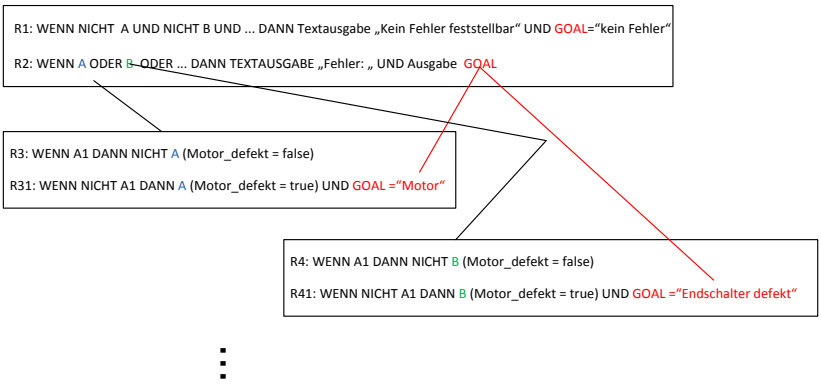

R1: WENN NICHT A UND NICHT B UND ... DANN Textausgabe „Kein Fehler feststellbar“ UND GOAL=“kein Fehler“
R2: WENN A ODER B ODER ... DANN TEXTAUSGABE „Fehler: „ UND Ausgabe GOAL
R3: WENN A1 DANN NICHT A (Motor_defekt = false)
R31: WENN NICHT A1 DANN A (Motor_defekt = true) UND GOAL =“Motor“
R4: WENN A1 DANN NICHT B (Motor_defekt = false)
R41: WENN NICHT A1 DANN B (Motor_defekt = true) UND GOAL =“Endschalter defekt“

Abbildung 4.32: *Schematische Darstellung der Goalbasierten-Verarbeitung*

4.6.3 Frameverarbeitung

Frameverarbeitung bietet sich nur an wenn große Objektnetze zu verarbeiten sind. So kann man überall dort wo man Bauteilnetze verwalten muss gut mit Frames arbeiten. Das Problem bei einer reinen Frameverarbeitung liegt in der Kommunikation mit dem Nutzer. Überall dort wo man andere Quellen zur Datenaufnahme ansprechen kann, etwa eine Steuerung, scheinen sie mir sinnvoll. So könnte in einem automatischen Diagnosesystem durchaus auf eine Fehlerquelle inferiert werden.
Die Frameverarbeitung lässt sich heute mit einer Objektorientierten Programmiersprache durchführen, wobei man immer im Gedächtnis behalten sollte, dass die Objektorientierte Programmierung maßgeblich durch die Frames angeregt wurden, aber dennoch nicht das Gleiche ist. Die Objektorientierte Programmierung ist heute viel mächtiger als das ursprüngliche Framekonzept, das ursprünglich nur bestimmtes Rollenverhalten abbilden sollte /Barr 81-82/.
Generell ist die Implementierung mit jeder Programmiersprache möglich, die Strukturen oder Records zulässt.
Das Verhalten des Systems wird weitgehend durch Trigger bestimmt. Diese stellen „sich selbst aktivierende“ Prozeduren dar, die in Aktion treten sobald das Objekt angefasst wird. Mit Hilfe dieser Trigger kann der Benutzer oder eine andere Schnittstelle zum Beispiel nach bestimmten Slotbelegungen befragt werden oder es andere Prozeduren aufgerufen werden.
Als Beispiel will ich erneut die schon im vorherigen Kapitel verwendete Bohrstation verwenden, die jetzt als Bauteilebaum dargestellt wird (siehe Abbildung 4.33).

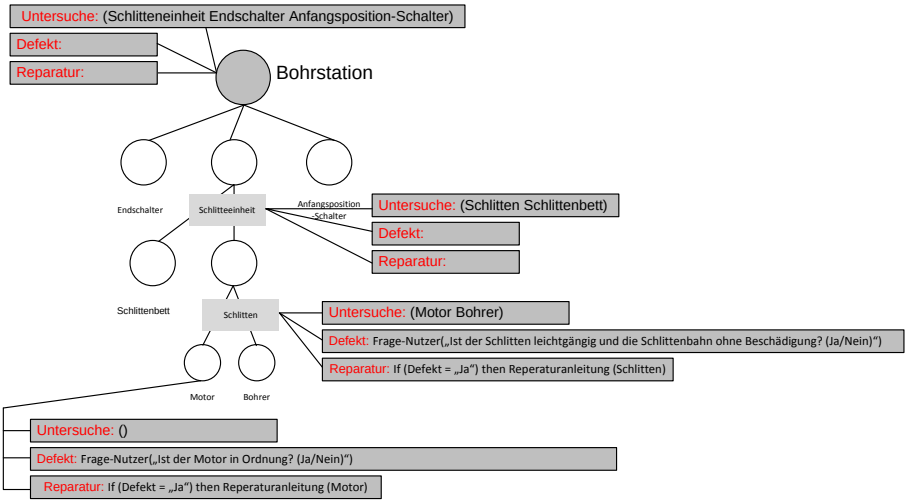


Abbildung 4.33: *Bauteilebaum-Darstellung der Bohrstation*

Die Problemlösung eines framebasierten Systems startet von einem ausgezeichneten Objekt (z. B. als „Root" bezeichnet). In unserem Beispiel sei dies die Bohrstation. Von diesem ausgezeichneten Objekt werden über Trigger die entsprechenden anderen Objekte ausgewählt. So könnte der Benutzer zum Beispiel gefragt werden, wo er den Fehler vermutet oder es könnten ähnliche Fragen gestellt werde, wie im vorherigen Kapitel in der Regelverarbeitung, um die zu untersuchenden Objekte auszusuchen.
Die ausgewählten Objekte werden meist in einer Liste zwischengespeichert, wo sie entsprechend einer vom Implementator festgelegten Strategie verarbeitet werden. Gebräuchlich ist hier die Tiefen- bzw. Breitensuche, die sich einfach aus der Verwaltung dieser Liste ergibt. Werden Objekte immer an den Anfang der Liste eingeordnet ergibt sich eine Tiefensuche, werden sie am Ende eingeordnet eine Breitensuche. Auch die Einordnung nach vorher vom KE festgelegten Prioritäten ist möglich.
Aus dieser Liste wird das erste Element genommen und dessen Trigger verarbeitet. Der Inferenzvorgang endet, wenn die Liste leer ist. Bei jedem untersuchten Objekt werden dabei bestimmte festgelegte Slots aktiviert.
Beispiel der Verarbeitung des Objekts „Bohrstation"

Im Beispiel werden die Objektslots UNTERSUCHE, DEFEKT und REPARATUR durch den Inferenzer verarbeitet. Der Problemlösungsvorgang wird über die Slotbelegung beeinflusst. Ist der UNTERSUCHE Slot leer kann sofort zum DEFEKT Slot übergegangen werden. Liefert dieser einen bestimmten Wert, so wird zum Slot REPARATUR fortgeschritten. Ist der UNTERSUCHE Slot allerdings nicht leer kann die Problemlösung:

- mit den angegebenen Objekten direkt fortfahren. Diese werden an den Anfang der Liste geschrieben, die vom Inferenzer sequentiell abgearbeitet wird. Dann muss das betrachtete Objekt zum Beispiel ans Ende der Liste vermerkt werden, um später nochmals die Slots DEFEKT und REPARATUR dieses Objekts untersuchen zu können; oder
- die Slots DEFEKT und REPARATUR des betrachteten Objekts werden direkt verarbeitet. Damit wird ein möglicher Fehler dieses Bauteils direkt gefunden, die Untersuchung aber dennoch fortgesetzt. Das betrachtete Objket kann dann aus der Liste entfernt werden.

Im Beispiel soll zur Übersichtlichkeit immer die zweite Variante verwendet werden.

1. Schritt

Das betrachtete Objekt ist **Bohrstation** (als Root angenommen)

Liste der zu untersuchenden Objekte (Bohrstation)

Der Trigger (hier mit Untersuche bezeichnet) der Bohrmaschine könnte nun alle untergeordneten Objekte in diese Liste einordnen und diese sukzessive abarbeiten.

Objekt Bohrstation

..

Untersuche : (Schlitteneinheit Endschalter Ausgangspositions-Schalter)

Diese werden in die Untersuchungsliste aufgenommen:

Liste der zu untersuchenden Objekte (Schlitteneinheit Endschalter Ausgangspositions-Schalter)

2. Schritt

Das betrachtete Objekt ist **Schlitteneinheit**
Objekt Schlitteneinheit
..
Untersuche : (Schlitten Schlittenbett)
Diese werden in die Untersuchungsliste aufgenommen, Schlitteneinheit daraus gestrichen:

Liste der zu untersuchenden Objekte
(~~Schlitteneinheit~~ Schlitten Schlittenbett Endschalter Ausgangspositions-Schalter)

3. Schritt

Das betrachtete Objekt ist **Schlitten**
Hier erfolgt die Untersuchung von Motor und Bohrer (diese werden in die Liste geschrieben).
Diese werden in die Untersuchungsliste aufgenommen:
Diese werden in die Untersuchungsliste aufgenommen, Schlitten daraus gestrichen:

Liste der zu untersuchenden Objekte
(~~Schlitten~~ Motor Bohrer Schlittenbett Endschalter Ausgangspositions-Schalter)

…

Nach deren Abspeicherung von Bohrer und Motor wird der Slot DEFEKT ausgeführt.
…
Der Nutzer wird „Ist der Schlitten leichtgängig und die Schlittenbahn ohne Beschädigung? (Ja/Nein)“ gefragt.
Etwa in der Form:
Defekt: Frage-Nutzer(„Ist der Schlitten leichtgängig und die Schlittenbahn ohne Beschädigung? (Ja/Nein)“)
Frage-Nutzer () ist dabei eine in der Dialogkomponente definierte Funktion die den entsprechenden Text ausstellt. Natürlich könnte dabei auch ein entsprechendes Medium, das ebenfalls an das Objekt gebunden ist, ausgestellt werden.
Reparatur: If (Defekt = „Ja“) then Reparaturanleitung (Schlitten)

In den weiteren Schritten werden dann noch die anderen Objekte entsprechend verarbeitet. Dadurch werden alle Fehler gefunden, falls der Nutzer nicht vorher abbricht. Die Inferenz endet wenn die Liste leer ist.

Liste der zu untersuchenden Objekte
(~~Motor~~ Bohrer Schlittenbett Endschalter Ausgangspositions-Schalter)

Liste der zu untersuchenden Objekte
(~~Bohrer~~ Schlittenbett Endschalter Ausgangspositions-Schalter)

...

Liste der zu untersuchenden Objekte
()

Die im Beispiel der Regelverarbeitung ebenfalls dargestellten Fehler Kühlflüssigkeit und Stromversorgung könnten (weil sie ja an keine Objekte gebunden sind) hier:

- durch Pseudo-Objekte verarbeitet werden (etwa Stromversorgung und Kühlwasserversorgung). Diese sind aber nicht Bestandteil der Bohrstation sondern liefern Vorbedingungen (siehe hierzu auch Kapitel 5) oder
- durch entsprechende Fragen an den Benutzer zum Beispiel direkt am obersten Objekt verarbeitet werden.

4.6.4 Verarbeitung innerhalb hybrider Systeme

Reine Regelbasierte Inferenz oder Framebasierte Systeme haben sich als unflexibel erwiesen. Die Objektorientierung einer Wissensbasis macht eine Modellierung möglich, die sich an der Realität orientiert, die Regelverarbeitung stellt hingegen eine Problemlösungsverhalten dar, die sehr ähnlich dem im Alltag verwendeten menschlichen Problemverhalten ist. Aus diesem Grund entwickelten sich die XPS zu hybriden Systemen, in den verschiedenen Wissensrepräsentationsformen miteinander verschmolzen.
Frames und Reglverarbeitung lassen sich einfach so miteinander kombinieren, dass die Regeln bzw. Regelgruppen über Slots an Objekte gebunden werden. Sie können dabei als Elemente verstanden werden, die wie die oben betrachteten Bauteile ihn einer Verarbeitungsliste verwaltet werden. Siehe hierzu auch Kapitel 5.4.

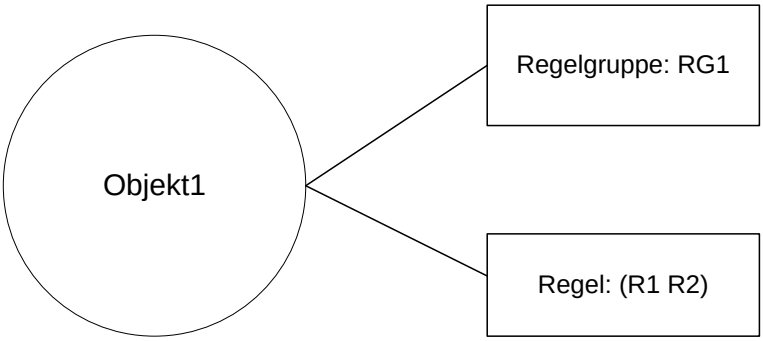


Abbildung 4.34: *Verbindung zwischen Frames und Regeln*

4.6.5 Verteilung der Problemlösung

Eine starr operierende Problemlösungskomponente ist für die meisten Bereiche zu unflexibel. Vielmehr wird das Inferenzwissen immer wichtiger und soll deshalb ebenfalls abgebildet werden. Daher lag es nahe den Problemlösungsvorgang in einzelne Schritte aufzuteilen, die vom KE an die verschiedenen Objekte in der Wissensbasis gebunden werden können.
Dies stellt mehr Anforderungen an das Knowledge Engineering , macht aber die ganz flexible Behandlung einzelner Objekte möglich. Die eigentliche Problemlösungskomponente stellt dabei ein klassisches Programmmodul dar, das die Prozeduren zur Verarbeitung der Inferenzschritte enthält. Diese werden direkt bei der Verarbeitung der Wissensbasis aufgerufen.
Inferenzschritte können dabei prinzipiell an alle Wissenselemente gebunden werden. So gibt es Schritte, die die Verarbeitung der verschiedenen Objekte, der Regelgruppen, Regeln und selbst Aussagen steuern. Innerhalb einer Wissensrepräsentationssprache werden diese einfach den verschiedenen Wissenselementen über bestimmte Codeworte zugeordnet (etwa UNTERSUCHE BAUTEIL „Schlitteneinheit“). Dieses Codewort wird dann vom Inferenzer entsprechend der implementierten Problemlösungs-Strategie verarbeitet (etwa in Form der oben angeführten Beispiele).

Auf diese Weise können auch Strategien wie Vorwärts- oder Rückwärtsverkettet einzelnen Regelgruppen, Regeln oder Aussagen zugeordnet werden. Der Inferenzer kann an bestimmten Stellen genau so vorgehen wie der Experte, der sein Lösungskonzept ja auch dem Problem anpasst.
Eine beispielhafte Wissensrepräsentation dieser Inferenzschritte wird in Kapitel 5 dargestellt.

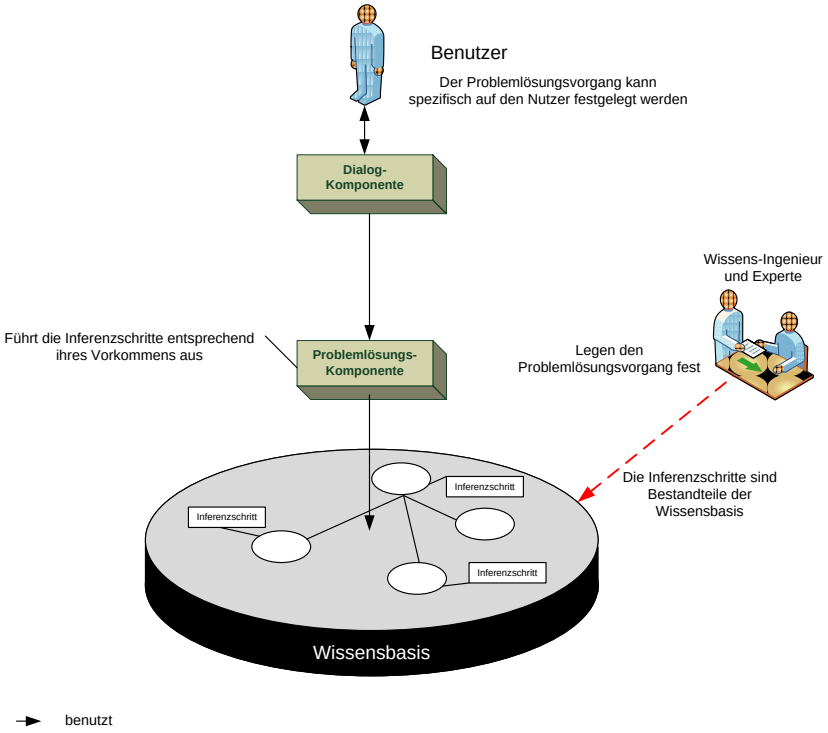


Abbildung 4.35: *Verteilung der Problemlösung auf verschiedene Inferenzschritte*

Eine mögliche Verarbeitungsstrategie der Inferenzschritte, die sich in der Praxis bewährt hat, stellt die Verwendung von Koroutinen in der Inferenz dar. Dabei werden die Inferenzschritte vom KE den verschiedenen Koroutinen zugeordnet, die dann verschiedenen Objekten zugeordnet werden. Anders als Subroutinen, die beim Wiederaufruf immer wieder an einer ganz speziellen Stelle mit der Arbeit beginnen, wird bei Koroutinen an der Stelle fortgefahren, an der die Routine verlassen wurde. Dies bedeutet, dass eine Koroutine nur einmal vollständig durchgearbeitet wird und dann beendet ist.
Die Koroutinen enthalten nun die Inferenzschritte, die nacheinander abgearbeitet werden. Spezielle Schritte ermöglichen es dem Inferenzmechanismus diese Koroutine an bestimmten Stellen zu verlassen und zu anderen Koroutinen zu verzweigen, die dann entweder an der Stelle fortgesetzt werden, an denen sie aufgegeben wurden, oder fall sie noch nicht angesprochen wurden mit dem ersten Inferenzschritt beginnen.
Koroutinen werden über ihre in einer Verarbeitungsliste verwalteten Objekte (etwa Bauteile) angesprochen. In der Verarbeitungsliste wird das an erster Stelle stehende Objekt ausgewählt und dessen Koroutine angesprungen. Wird die Koroutine aufgegeben, so wird das entsprechende Element ans Ende der Liste geschrieben. Falls sie ganz verarbeitet wurde, wird das Objket aus der Liste gelöscht.

Beispiel:

Die Bauteile Umsetzer, Regal-Lager und Fräse sollen untersucht werden. Dazu sind sie in die Verarbeitungsliste des Inferenzeres geschrieben worden. Die Koroutinen des esreten Bauteils (hier Umsetzer) wird angesprochen. Abbildung 4.36 zeigt die Situation in der die Koroutine aktiv ist und nachdem sie sich selbst aufgegeben hat (der entsprechende Inferenzschritt heißt hier Hypothesenverfolgung). Die Koroutinen wird aufgegeben dann und an das Ende der Liste geschrieben.
Die nächste zu aktivierende Koroutine ist dann die des Regal-Lagers, mit deren Verarbeitung an der Stelle fortgefahren wird an der sie sich durch den entsprechenden Inferenzschritt aufgegeben hatte.

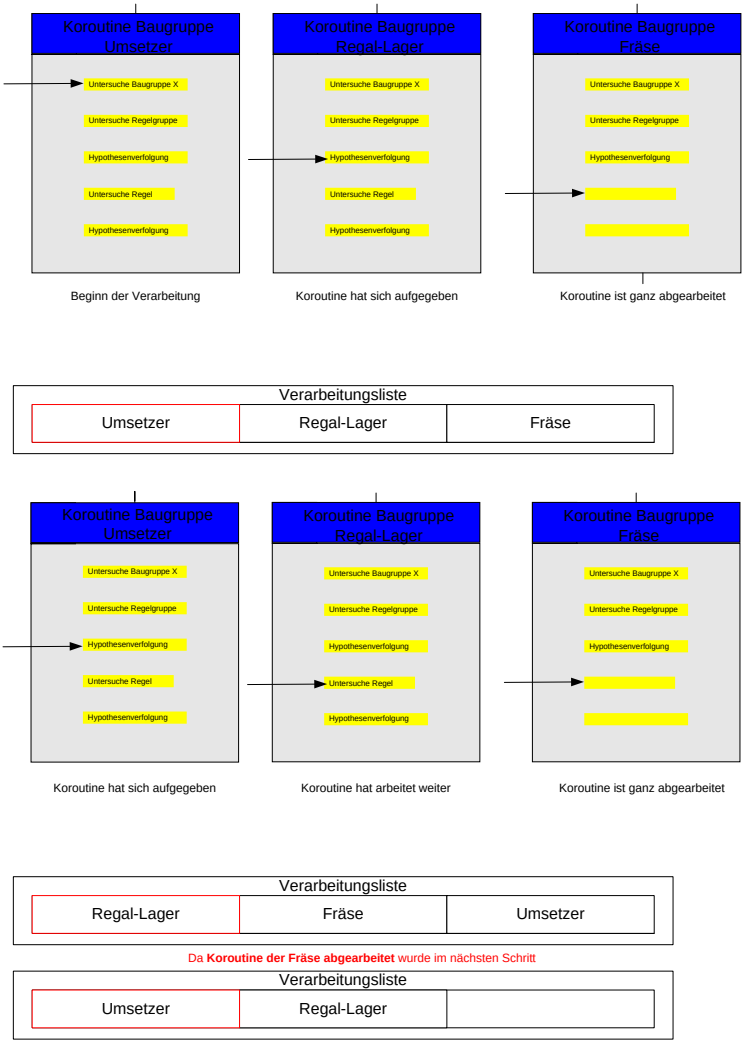


Abbildung 4.36: *Koroutinenverarbeitung*

Wird diese Koroutine aufgegeben, so wird zunächst die Koroutine der Fräse angesprungen und da festgestellt wird, dass die Koroutine vollständig abgearbeitet wurde wird sie aus der Verarbeitungsliste gelöscht und mit der Koroutien des ersten Bauteils der Liste fortgefahren.
Die Inferenz endet, wenn kein die Verarbeitungsliste leer ist. Zum Start des Systems wird eine ganz spezielle Objket in diese Liste geschrieben (etwa die des Root-Objekts) oder dem Benutzer wird ein Objekt zur Auswahl angeboten. Im Expertensystem-Shell SIGMA konnte der Startpunkt auch über ein von der SPS geliefertes Signal erfolgen. Dazu wurde über die Schnittstellekomponente mittels Polling auf die Speicherung zugegriffen.
Der Inhalt der Koroutinen und damit auch der Zeitpunkt wann sie sich aufgibt wird vom KE in Zusammenarbeit mit dem Experten festgelegt.
In der Diagnostik wird die Verarbeitungsliste auch als Hypothesenfokus bezeichnet, weil dort die verschiedenen Hypothesen gesammelt werden, die man im Laufe der Problemlösung untersuchen möchte. Der Name Hypothesenverfolgung weist hier darauf hin, dass die nächste in diesem Fokus stehende Hypothese nun untersucht werden soll. Die Verarbeitungsliste enthält natürlich alle Objekte, die untersucht werden sollen und nicht nur die Koroutinen. Siehe hierzu auch Kapitel 5.

4.6.3 Verarbeitung von unsicherem Wissen

Um unsicheres Wissen zu verarbeiten, können die verschiedenen Wissenselemente mit Wichtungsfaktoren verbunden werden. Diese Certainty Factors (engl. Certainty für Gewissheit/Langenscheid 97/) können auf Wahrscheinlichkeiten, auf Fuzzy-Logik oder auf intuitiven Bewertungskonzepten beruhen. Beispielhaft sei hier die Verwendung eines intuitiven Konzepts bei MYCIN kurz vorgestellt. Die Verwendung von Wahrscheinlichkeiten ist deshalb problematisch, weil zu vielen Bereichen keine empirischen Untersuchungen vorliegen, die Verwendung von Fuzzy-Logik unterliegt den gleichen Einschränkungen wie die der intuitiven Konzepte, die unten angegeben sind.
Zum ersten Mal aufgetreten sind diese schon dort als CF bezeichneten Faktoren bei der Entwicklung von MYCIN, wo sie ein intuitives Konzept darstellen /Reif 00/.
Man unterscheidet bei MYCIN zwei verschiedene Arten von Certainty Factors:

- Dynamische Certainty Factors

  Durch diese Wichtungen werden Fakten, die nicht als definitiv wahr oder falsch angenommen werden können quantitativ bewertet. Dabei wird zur Wichtung ein Intervall zwischen -1 und 1 verwendet, wobei -1 definitiv falsch und 1 definitiv richtig bezeichnet.
  Die Bewertungen basieren dabei auf einem bestimmten Hintergrundwissen, welches hier als Evidenz (unmittelbare Einsichtigkeit, Gewissheit / Duden 10/) bezeichnet wird. CF(F|E) beschreibt in diesem Zusammenhang zum Beispiel einen Certainty Factor auf Basis der Evidenz E. Durch den Problemlösungsprozess kann sich dieses Hintergrundwissen ändern, weil ja durch neue "Einsichten" Faktoren abgeleitet werden können, womit sich auch die Certainty Factors, die den Faktoren zugeordnet sind, verändern können. Aufgrund der Möglichkeit der Veränderung spricht man hier von dynamischen Certainty Factors.

- Feste Certainty Factors

  Mit Hilfe von fest zugeordneten Certainty Factors legt der KE eine unischere Abhängigkeit zwischen der Prämisse und der Konklusion einer Regel fest.

Beispiel:

WENN	DANN
Schlitteneinheit="schwergängig" UND Bettbahn = „zerkratzt"	VERSCHMUTZUNG_DURCH_VERKLEMMTE_SPÄNE Verdacht (0.7)

Die Regel beschreibt dabei die bedingte Sicherheit, dass die Ursache für einen schwergängige Schlitteneinheit bei gleichzeitig festgestellten Kratzern auf der Bettbahn, auf der der Schlitten gleitet, ein verklemmter Span ist, der sich während des Spanvorgangs dort festgesetzt hat. Formal kann diese bedingte Sicherheit der Hypothese H unter der Voraussetzung, dass die Faktoren F1 (Schlitteneinheit="schwergängig") und F2(Bettbahn = „zerkratzt") wahr sind so dargestellt werden: CF(H|F1 ^ F2).

Unter Verweis auf /Heinsohn 99/ beschreibt Reif folgende mögliche Arbeitsschritte innerhalb eines XPS, welches mit Certainty Factors arbeitet:

- die dynamischen CF werden vom Benutzer eingegeben,
- bei aus mehreren Literalen zusammengesetzten Prämissen wird ein CF-Faktor für die Prämisse errechnet,
- aus dem CF-Faktor der Prämisse wird unter Einbeziehung des fest vorgegebenen CF-Faktors eine gesamt CF für die Hypothese gebildet,
- da die Hypothese aber durch mehrere Regeln behandelt werden kann, ergeben sich eventuell mehrere verschiedene CF einer Hypothese, die dann wieder zusammenfasst werden müssen dieser Vorgang wird als parallele Kombination bezeichnet.

Wie man an diesen Ausführungen leicht sieht ergeben sich hierbei einige Probleme:

- die Eingabe von Faktoren durch den Benutzer ist sehr willkürlich, d.h. die Problemlösung und das gefundene Ergebnis werden abhängig von Nutzer des Systems. Damit kann die Qualität der gefundenen Lösungen stark streuen;
- wie ich aus eigener Erfahrung weiß, sind normale Nutzer bei der Frage nach einem Faktor meist vollständig überfordert und werden dadurch bei der Handhabung des Systems verunsichert;
- selbst die fest zugeordneten CF sind sehr willkürlich angeordnet. Wie will ein Experte, der sich nicht auf fundierte empirische Studien stützen kann, beurteilen ob ein CF von 0,5 oder doch eher 0,47 genommen werden soll.

Bei der Entwicklung unserer Expertensysteme /Döll 89/ mussten wir aus vorgenannten Gründen die vorgesehene Verwendung von CF aufgeben.
Auch die Einführung zweier weitere Maßeinheiten in MYCIN, die als Maß für das Vertrauen MB (Measure of Belief) und MD (Measure of Disbelief) bezeichnet werden, ändert für mich an den vorgemachten Kritikpunkten nichts. Hier müssen sogar zwei individuell einzuschätzende Werte eingegeben werden, was bei den Nutzern unserer XPS einfach nicht möglich gewesen wäre.
Die beiden Maßeinheiten werden nach /Reif 00, S. 127/ und /Heinsohn 99/ wie folgt definiert:

MB(h|E) = x:	„im Falle der Evidenz E wächst das Vertrauen, dass h wahr ist um x“

MD(h|E) = x:	„im Falle der Evidenz E wächst das Misstrauen, dass h wahr ist um x“

Aus diesen Werten wird dann der eigentliche Certainty Faktor wie folgt berechnet:

$$CF(h|E) = \frac{MB(h|E) - MD(h|E)}{1 - \min\{MB(h|E), MD(h|E)\}}$$

4.6.4 Rücknahme von Eingaben und Belegungen

Um das System wieder auf einen fest definierten Stand zu bringen kann man vorsehen, dass:

- die Wissensbasis vollständig in einen Anfangszustand zurückgesetzt wird und
- das einzelne Eingaben vom Benutzer wieder zurückgenommen werden können.

Das Rücksetzen der vollständigen Wissensbasis kann einfach so gelöst werden, dass die ursprüngliche Wissensbasis erneut geladen wird. Damit werden alle Wissenselemente in den Initialzustand versetzt.
Die Rücknahme von Belegungen wird vom Nutzer zwar während des Problemlösungsvorgangs durchgeführt, ist intern aber der Erklärungskomponente zuzuordnen, weil diese die entsprechenden Listen verwaltet, die dies erst ermöglicht.
Diese Listen sind notwendig, weil mehr Belegungen stattfinden als es der Benutzer bemerkt, weil auch über das Feuern von Regeln entsprechende Wissenselemente wie Aussagen, Bauteile, Baugruppen, Koroutinen, Regeln und Regelgruppen manipuliert werden können. Um dem Benutzer den Inferenzvorgang erklären zu können, werden für jedes manipulierbare Element der Wissensbasis entsprechende Einträge in einer Liste notwendig. Diese können dann bei der Zurücknahme von Belegungen verwendet werden. Da

bei der Zurücknahme einer Belegung auch alle daraus abgeleiteten Belegungen wieder zurückgenommen werden müssen, ist der Systembestandteil, der die Belegung vorgenommen hat und der Kontext mit zu vermerken. Der Benutzer muss bei der Rücknahme in die gleiche Situation geführt werden, damit er auch erkennt, was die Rücknahme für Konsequenzen hat. Die Rücknahme einer Aussage kann zum Beispiel auch über die verschiedenen Regelstrategien Auswirkungen auf andere Regeln und Aussagen haben.
Lässt das System zu, dass jede beliebige Belegung ohne Beachtung der Eingabereihenfolge zurückgenommen werden kann, können sich leicht für den Nutzer Systemzustände ergeben, die dieser nicht mehr durchblickt, weil damit womöglich immer nur Teile der Wissensbasis verändert werden. Das schrittweise Zurückgehen im Problemlösungsvorgang ist hier eine für das System leichter durchführbare und für den Benutzer verständlichere Lösung.
Eine in der Literatur erwähnte Möglichkeit zur Zurücknahme von Belegungen stellt ein Assumption Based Truth Maintenance System (ATMS) dar/De Kleer 86/.

Beispiel:

Die Belegung der Aussage Schlitteneinheit_OK wird zurückgenommen, weil doch noch ein Fehler gefunden wurde. Alle Regeln, in denen Schlitteneinheit_OK in der Prämisse steht, müssen nun neu erfragt werden, wobei zu entscheiden ist, ob auch Fakten neu belegt werden sollten, die der Benutzer gar nicht ursprünglich zurücknehmen wollte, die aber im Zusammenhang mit der betrachteten Aussage stehen. Alle in den Konklusionen dieser Regeln belegten Fakten müssen zurückgenommen werden, falls die Prämisse bei der Neubefragung nicht feuert.
Führte das Ausführen der Konklusion zu Belegungen der Verarbeitungsliste, müssen die eingefügten Bestandteile wieder entfernt werden. Da diese Konklusions-Aussagen ihrerseits wieder in Prämissen vorkommen können, muss der Vorgang so lange wiederholt werden, bis alle betroffenen Regeln verarbeitet wurden.
Werden innerhalb der Inferenz Strategien verwendet, etwa Koroutinen, so müssen diese eventuell wieder angepasst werden. Sich aus der Verarbeitung ergebende neu zu untersuchende Objekte müssen zum Beispiel in die Verarbeitungsliste übernommen werden. Die Verfolgung der entsprechenden Strategien muss bis zur endgültigen Abarbeitung der Zurücknahme ausgesetzt werden.

Das Verhalten eines solchen Systems kann für den Benutzer befremdlich wirken, problematisch sind dabei auch Belegungen durch eine Schnittstellen-Komponente, etwa durch eine Steuerung, deren Zustand sich möglicherweise während der Problemlösung verändert haben kann.

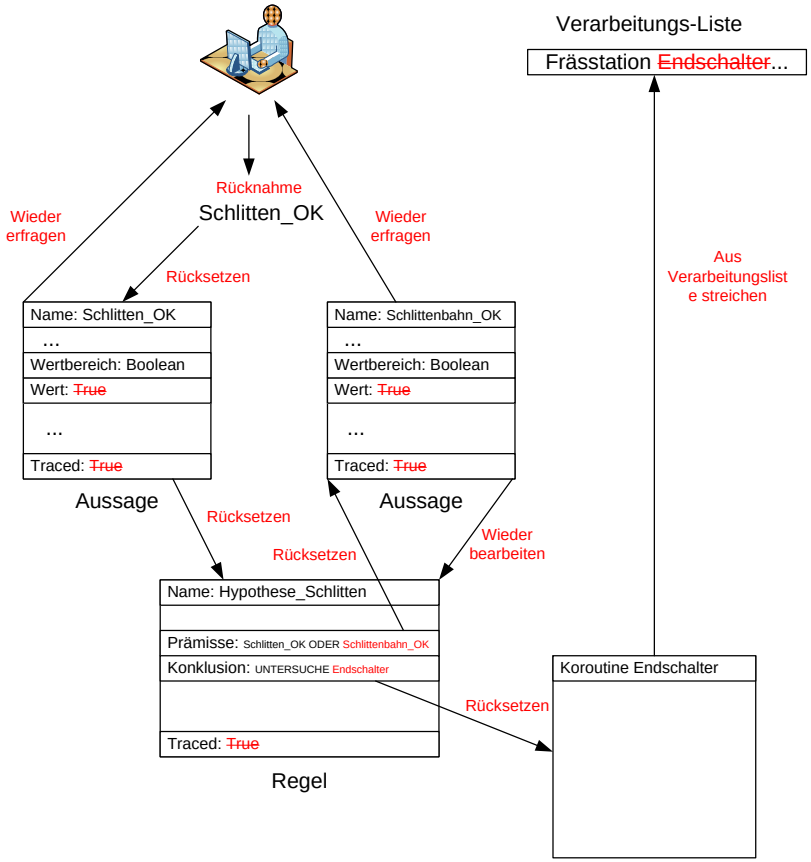


Abbildung 4.37: *Rücknahme von Belegungen (Beispiel Bohrstation)*

4.7 Problemumfeld

Im Folgenden möchte ich kurz auf einige Aspekte eingehen, die mir wichtig zur Abrundung des Themas sind und die innerhalb des vorliegenden Buchs nicht erschöpfend behandelt werden können.

4.7.1 Menschliche Problemlösung

Menschen haben im Verlaufe Ihrer Entwicklung Mechanismen entwickelt, wie Probleme strukturieren und behandelt werden können. Die Formale Beschreibung dieser Schlussregeln geht auf Aristoteles zurück und wird zum Beispiel in /Böhme 81/ und /Hartmann 09/ ausführlicher beschrieben.
Im Bereich der Künstlichen Intelligenz identifiziert /Knemeyer 94/ mit Bezug auf weitere Literaturstellen folgende vier hauptsächlich eingesetzten Inferenz-Klassen:

- Deduktive Inferenz
  Dabei werden spezielle Fakten durch allgemeingültige Gesetzmäßigkeiten hergeleitet.

- Induktive Inferenz
  Dabei wird aus Beobachtungen auf eine allgemeingütige Gesetzmäßigkeiten geschlossen.

- Statistische Inferenz
  Hier wird ein empirisch gestützter Fakt oder eine wahrscheinliche Schlussfolgerung verwendet.

- Analoge Inferenz
  Dabei wird versucht von einer bekannte Analogie auf den betrachteten Sachverhalt zu schließen.

Knemeyer untersucht im folgenden, wie diese von Menschen häufig verwendeten Mechanismen auch im Bereich der XPS eingesetzt werden können, weil das Schlussfolgern im Mittelpunkt jeden intelligenten Verhaltens steht und somit auch in den künstlich intelligenten Systemen von zentraler Bedeutung ist.

Als allgemeines Inferenzschemata versteht er, neben den vorher genannten, auch die aus der Deduktion abgeleitete Abduktion.

Deduktion
Die Deduktion wird mit Hilfe der Abtrennungsregel (modus ponens) definiert. Sie ist eine der am meisten in der XPS-Technologie verwendeten Schemata und hat folgenden logisch formalen Aufbau:

Wenn A wahr ist,	A	
und A, B impliziert,	A →	B
so wird geschlossen,		
dass auch B wahr ist		B

Abbildung 4.38: *logischer Aufbau der Deduktion*

Abduktion
Die Abduktion stellt keinen korrekten deduktiven Schluss dar, wie mit dem folgenden sehr oft zitierten Beispiel gezeigt werden soll:

Wenn es geregnet hat, ist die Strasse naß,	A →	B
Die Strasse ist naß,	B	
also		
hat es geregnet		A

Abbildung 4.39: *logischer Aufbau der Abduktion*

Dies ist kein korrekter deduktiver Schluss, weil die Straße ja auch durch einen anderen Faktor nass geworden sein kann, etwa durch die Straßenreinigung

Bedeutung hat die Abduktion als Abbildung des „gesunden Menschenverstandes“ („common sense reasoning“). Knemeyer formt die Regel wie folgt um und nennt die „quasi deduktiv“:

Wenn die Straße nass ist, hat es meist geregnet.	B → A
Die Straße ist nass	B
also hat es wohl geregnet	A

Abbildung 4.40: *Abduktion als gesunder Menschenverstand*

In der Literatur wird die Verwendung der Abduktion in wissensbasierten Systemen durchaus kontrovers diskutiert /Reichertz 94/.

Induktion
Innerhalb der Induktion schließt man von zwei oder mehr Beobachtungen auf eine allgemeine Regel. Dieses Beobachten und Verallgemeinern kann dabei als Lernen verstanden werden:

Martin ist ein Schwan und Martin ist weiß	S_{Martin} UND W_{Martin}
Volker ist ein Schwan und Volker ist weiß	S_{Volker} UND W_{Volker}
Usw.	…
Wenn x ein Schwan ist, dann ist x weiß	$S_x \rightarrow W_x$
Oder einfach: „Alle Schwäme sind weiß“.	S_x : x ist ein Schwan W_x: x ist Weiß

Abbildung 4.41: *logische Darstellung der Induktion*

Unsere Beobachtung hat Auswirkungen auf unser Lernen. Machen wir uns aber vielleicht an dieser Stelle klar, dass vieles was wir zu wissen meinen, etwa

die gleiche Qualität hat wie die Annahme, dass alle Schwäne weiß sind. Tipp: es gibt auch schwarze Schwäne.
Induktive Inferenz wird nach meiner Kenntnis nur sehr selten in XPS eingesetzt.

Statistische Inferenz
Statistisches Inferieren macht empirische Untersuchungen über den betrachteten Fall notwendig. Ohne entsprechende Daten kann man also nicht mit Wahrscheinlichkeiten schließen. Dies bewirkt, dass diese Schlußmechanismen nur auf eng begrenzte Anwendungsfälle anwendbar sind. Werden hier aber subjektive Annahmen als Wahrscheinlichkeiten angegeben, verliert die durch das System gefundene Problemlösung an Seriosität.

In Mallorca ist das Wetter oft schön (Wenn man Urlaub auf Mallorca macht, hat man oft schönes Wetter)	$U_x \rightarrow W_{Mallorca}\ (P_y)$
Ich mache Urlaub auf Mallorca	U_{ich}
Das Wetter wird wohl auch während meines Urlaubs schön sein (Ich habe wohl schönes Wetter)	$W_{Mallorca}\ (0.7)$

U_x : x macht Urlaub auf Mallorca
W_x : das Wetter ist in x schön
P_y : mit y Wahrscheinlichkeit

Abbildung 4.42: *Beispiel für statistische Inferenz*

Das Beispiel entspricht einer Beobachtung des Urlaubers. Der Schluss kann als common sense reasoning verstanden werden. Die in Klammer umgeformte Schreibweise soll verdeutlichen, dass es sich hier um eine Abduktion handelt. Mit Formulierungen wie oft und wohl wird eine Unsicherheit ausgedrückt, weil Erfahrung noch keine Gewissheit ist. (siehe hierzu auch Kapitel 4.6.x Certainty Factors).

Analog Schlüsse
Analog Schlüsse stellen die Übertragung einer Regel eines bekannten Anwendungsbereichs auf einen anderen Bereich mit ähnlicher Struktur dar, wobei die Gültigkeit der Regel auch in diesem Bereich angenommen wird. Dies ist damit im strengen Sinn kein logischer Schluss.

Wenn die Anzahl der Baumstämme, die man auf einem Fluss transportiert zu groß ist, dann stauen sich die Baumstämme an Engstellen	$A_{Baumstämme}$ UND $F_{Baumstämme}$ UND ($A_{Baumstämme} > F_{Baumstämme}$) → $S_{Baumstämme}$
Wenn die Anzahl der Pakete, die man in einem Rechnernetz transportiert zu groß ist, dann stauen sich die Pakete an den Stellen, an denen zu wenig Stauraum ist (Puffer)	A_{Pakete} UND F_{Pakete} UND ($A_{Pakete} > F_{Pakete}$) → S_{Pakate}
	A_x : Anzahl der x im Fluss/Rechnernetz F_x : der Fluss/Rechnernetz kann maximal von x aufnehmen S_x : der Fluss/Rechnernetz wird durch x gestaut

Abbildung 4.43: *Beispiel für Analog-Schlüsse*

Diese Strategie ist sehr schwer in XPS umzusetzen, weil jede Analogie nur mit Einschränkungen gelten kann („Jedes Beispiel hinkt!"). Das Gebiet des Maschinellen Lernens beschäftigt sich auch mit Lernen aus Analogien /Michalski 83/ /Mitchel 97//Michie 94//Alpaydin 08//Morik 93//Morik 05//wiki 10 k/

4.7.2 Fallbasiertes Problemlösen

Ein weiteres Modell für menschliches Problemlösen stellt das Fallbasierte Schließen dar (Case-Based Reasoning). Dieses geht von Erfahrungswissen aus, wobei die Erfahrung in Form von Fallbeispielen abgebildet wird. Auf diese „bekannten" Beispiele wird dann bei der Problemlösung zugegriffen.
Damit kann Erfahrungswissen, dass auf gelernte Beispiele referiert, in die XPS-Technologie einbezogen werden.
/Althoff 91/ beschreibt einen Fall als ein in der Realität vorkommendes Beispiel. Damit ein Fall als Beispiel verwendbar ist, muss dabei von zeitlich und räumlich eingrenzbaren Ereignisse bzw. Vorgängen abstrahiert werden. Diese Art der Wissensrepräsentation wird nach /Strube 89/ auch als episodisches Wissen bezeichnet.
Über die formale Beschreibung eines Falls besteht nach Althoff Uneinigkeit, er verwendet die für mich einsichtige folgende Definition:

„Ein Fall F ist gegeben als ein Tripel (P,R,L) mit einer Problembeschreibung P, einer Rechtfertigung R und einer Lösung L." /Althoff 91, S.2/

die auf /Veloso 89/ zurückgeht. Der Lösungsweg wird dabei als Rechtfertigung verstanden.

Beispiel eines Falls innerhalb der Diagnostik:

Problembeschreibung:	Die Beschreibung der verschiedenen Symptome, die in der Situation auftreten (Symptomatik).
Lösung:	die gestellte Diagnose
Rechtfertigung:	die Folge der Problembearbeitungsschritte, die letztlich zur Diagnose führen.

Innerhalb einer Expertensystem-Sitzung kann damit die Eingabe des Nutzers als Beschreibung der Symptomatik, die Diagnose als Lösung und die, durch die Erklärungskomponente gelieferte, Reihenfolge des Problemlösungsvorgangs als Rechtfertigung angesehen werden.

Die Fälle lassen sich nach Schwierigkeitsgraden der anpassbaren Fälle unterteilen. Neben den normal verwendbaren werden:

- Leicht anpassbare, sogenannte *Softcases*, *Routinefälle* oder *Fälle mit kleiner Variation und*
- *Hardcases*, also *Fälle mit großer Variation*

beschrieben.

Das Problem der Routinefälle in XPS ist, dass diese vom Benutzer aber vielleicht auch ohne System gelöst werden können, obwohl sie ebenfalls abgebildet werden müssen und einen großen Zeitaufwand bei der Wissensbasis-Erstellung benötigen.

Das Problem der Hardcases ist hingegen, dass zu ihrer Verarbeitung detailliertes Hintergrundwissen oder großes Common-Sense-Wissen notwendig ist.

Fallbasierte Systeme repräsentieren die Gesamtheit der Fälle in einem Fallgedächtnis (case memory). Zur Problemlösung müssen dann die Fälle aus diesem Gedächtnis „erinnert“ und bereitgestellt werden(retrievel), die dem untersuchten Problem ähnlich sind. Damit kann die fallbasierte Problemlösung als ein Spezialfall des analogen Schließens betrachtet werden.

Für die Beschreibung verschiedener Systeme, die fallbasiertes Problemlösung verwenden, sei an dieser Stelle nochmals auf /Althoff 91/ verwiesen.

4.7.3 Unterschiede zwischen XPS und DB

Durch die Trennung von zu verarbeitenden Elementen und der Verarbeitung selbst, erfolgt in Systemen eine Produktivitätssteigerung, weil damit sowohl die Präsentation der Elemente wie die Strategie der Verarbeitung optimiert werden kann. In Datenbanken erfolgt die Trennung zwischen Daten und Datenverarbeitung in XPS zwischen Wissen und Wissensverarbeitung. XPS unterscheiden sich dabei aber von den Datenbaken dadurch, dass sie ihre verschiedenen Repräsentationsformen (etwa Regeln und Frames) ausnutzen, um daraus auch neues Wissen zu erzeugen. /Mörler 84/
Expertensysteme benutzen zur Inferenz zum Teil Heuristik, sind selbstlernend und können dem Benutzer ihren Problemlösungsprozess auch erklären. /Zabel 00/während die frühen Datenbanken die Eingabe von streng strukturierten Daten(Zeichen, Zahlen) voraussetzten und daher streng numerisch formalisiert waren/Pietzka 95/.
Die neuere Entwicklung innerhalb der Datenbanktechnologie erweitert das relationale Datenbankmodell hingegen zu sogenannten „deduktiven" Datenbanken, welche Deduktionsregeln verwenden, um Wissen aus den abgespeicherten Daten zu extrahieren. Als Regelsprache wird dabei Datalog verwendet, was aus den beiden Begriffen Data und Prolog zusammengesetzt sein soll./Spreckelsen 08/ /wiki 10l//Rieger 04/

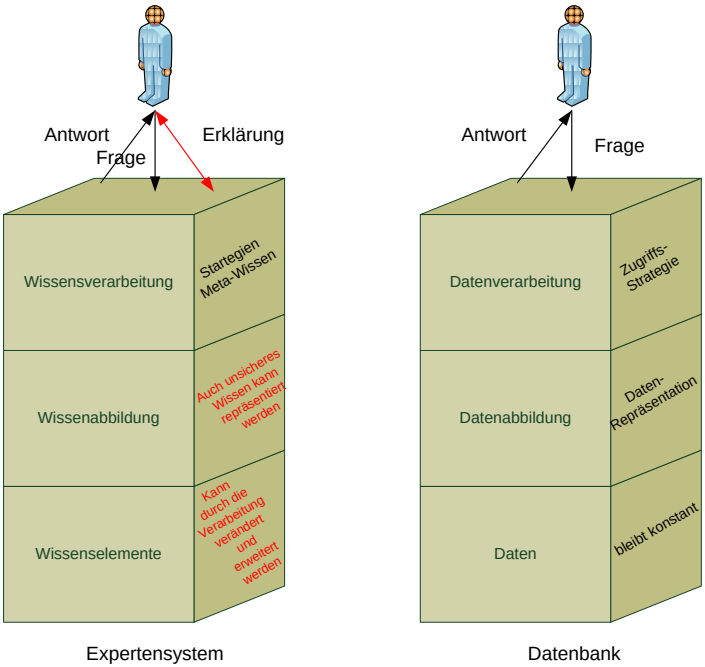


Abbildung 4.44: *Struktur der Verarbeitung von XPS und Datenbank*

4.7.4 Beispiel eines verteilten XPS

Die in diesem Buch beschriebenen Repräsentationsformen wie Regeln und Frames können auch auf verschiedene Rechner verteilt werden und zusammen an einer bestimmten Aufgabe arbeiten. Auch wenn die Wissenschaft in den letzten 20 Jahren natürlich sehr weit fortgeschritten ist, möchte ich aus Anschaulichkeitsgründen hier eine kleine ältere Anwendung vorstellen.
Das verteilte Wissensbasierte System entstand im Rahmen meiner Dissertation /Hartmann 92/ und bestand aus den Komponenten:

- Interpretations-Komponente
  Zur Interpretation von Daten, die zwischen Rechnern ausgetauscht werden. Damit konnte zum Beispiel Daten einer CAD-Darstellung in CAE etc. umgewandelt und angepasst werden.

- Statistik-Komponente
  Diese führte Statistiken, die dazu benutzt werden konnten zum Beispiel das Verhalten der beteiligten Maschinen zu dokumentieren.

- Diagnose-Komponente
  Die auf die ermittelten Daten der Statistik-Komponente und Netzwerk-Komponente zugreifen und daraus Fehler in den Komponenten oder im System feststellen konnte.

- Netzwerk-Komponente
  Diese war dafür zuständig das Netzwerk so zu verwalten, dass ein optimales Arbeiten möglich war.

Die Wissensrepräsentation musste dabei auf die speziellen Verarbeitungsabsichten angepasst werden. Hierzu wurde eine eigenständige Sprache entwickelt, die auf den entsprechenden Assemblern der verwendeten Computersystemen (CT, AT, Amiga und Atari) aufsetzte und daher eine Nebenläufigkeit zuließ. Auf den einzelnen Netzrechnern konnte jeweils eine der Komponenten aktiviert sein. Dabei konnte die gleiche Komponente auch mehrfach im System aktiviert sein, um etwa innerhalb der Diagnostik bestimmte Netzwerkobjekte zu diagnostizieren. Heute würde man die hier angedeuteten Komponenten wahrscheinlich als Netzwerkagenten bezeichnen.

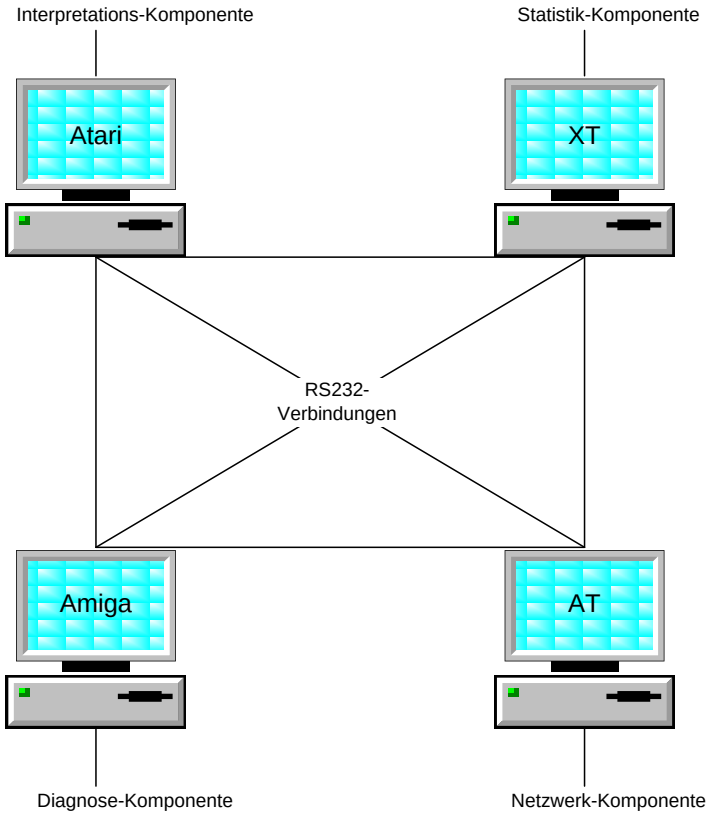


Abbildung 4.45: *Struktur des verteilten XPS-Systems*

Zum Überblick über die neuste Entwicklung im Bereich der verteilten wissensbasierten Systeme möchte ich auf /Buhl 96//Brauer 91/verweisen.

5 Das rote Wunder

In diesem Kapitel möchte ich ein studentisches Projekt beschreiben, welches ich im Rahmen der Vorlesung „wissensbasierte Systeme“ im Masterstudiengang MIKS von den Studierenden bearbeiten lasse. Ziel ist es dabei ein Expertensystem zu realisieren, welches die technische Diagnose einer Transferstraße unterstützt. Dabei wird eine kleine Transferstraße verwendet welche zur Ausrüstung des Prozess-Datenverarbeitungs-Labor des Fachbereichs gehört.

Da die Entwicklung eines Expertensystems eine langwierige Angelegenheit ist, beschäftigt sich gegenwärtig das dritte Semester mit diesem Projekt, wobei die Studierenden die Ergebnisse ihrer Vorgänger weiterführen. Die hier verwendete Wissensrepräsentationssprache wurde von den Studierenden nach meinen Vorgaben entwickelt, die verwendeten Beispiel-Wissensbasen sind Ergebnis der Studierenden. Die erste Generation der Studierenden hat dabei auch das Privileg den Namen des Gesamtprojekts zu bestimmen.

5.1 Vorgehen innerhalb der Diagnose

Da sich das System mit der technischen Diagnose beschäftigt, möchte ich zu Beginn einen kurzen Blick auf den Diagnosevorgang allgemein werfen.

Innerhalb der technischen Diagnose erhält man Symptomen, aus denen man Hypothesen für den möglichen Fehler ableitet. Diese Hypothesen versucht man dann durch Befragen des Nutzers oder durch Abfragen der Steuerung zu verifizieren.

Dieses Vorgehen wird auch in der Medizin durchgeführt, wo der Arzt von einer Reihe von Hypothesen zu einer Krankheit durch Fragen an den Patienten die wahrscheinlichste dieser Vermutungen herauszufinden sucht. Die Technik wird als Differential- oder Ausschlussdiagnose bezeichnet. /Cawsey 03/

Bei der Untersuchung werden, ausgehend aus den vorgegebenen Symptomen, alle anderen Erklärungen ausgeschlossen, weil das Symptombild meist nicht nur auf einen ganz bestimmten Fehler zutrifft sondern unterschiedliche Ursachen offen lässt. /wiki 10h/

Allgemein kann man die medizinische Behandlung in eine Diagnose- und eine Therapieschleife einteilen. Innerhalb einer Diagnoseschleife versucht der Arzt durch die Wiederholung der Schritte

- Symptomsuche: Auswahl von Möglichkeiten der Bestimmung von Symptomen,
- Symptomerhebung: Anwendung der ausgewählten Verfahren,
- Symptomerkennung: Auswertung der Ergebnisse dieser Anwendung,
- Diagnostik: Interpretation der gefundenen Symptome

die Krankheit des Patienten zu erkennen. /Uni Würzburg 04/
In der Theraphieschleife, die aus

- Therapiebestimmung Aufgrund der Diagnose wird eine Auswahl von Heilungsmöglichkeiten ausgewählt. Dabei ist es möglich, dass wieder in die Diagnoseschleife gesprungen werden muss, wenn es zur Auswahl notwendig wird;
- Theraphiedurchführung: die eigentliche Anwendung einer Therapie und
- Therapieüberwachung die Überprüfung ob die Behandlung den gewünschten Erfolg hatte

besteht, versucht man dann die gefundene Krankheit zu heilen. Gelingt dies nicht, muss der Arzt wieder in die Diagnoseschleife springen und eine weitere Hypothese für den gefundenen Symptomzustand suchen. /Uni Würzburg 04/
Die technische Diagnose geht dabei sehr ähnlich vor, wobei hier nur die Reparatur den Begriff Therapie ersetzt.
Eine etwas andere Betrachtungsweise stellt /Wiegand 04/mit der Modellorientierten Diagnose vor. Die Modellbasierte Diagnose unterteilt sich dabei in die Wirkungsanalyse (auch als Fehlerbaumanalyse bezeichnet) und die Ursachenanalyse (auch als Ereignisablaufanalyse bezeichnet).
Ausgehend von einem Fehlerzustand, der durch die Symptome beschrieben wird, versucht die Ursachenanalyse dessen Ursachen zu finden. Dabei wird der gesamte Entstehungsverlauf der Fehler bis zu ihrer Ursache untersucht.
Die Symptomerkennung erfolgt hier durch den Vergleich des zu untersuchenden Zustands mit dem Normalzustand und mit als Fall abgespeicherten anderen Fehlerzuständen. Zur Eingrenzung der Ursachen werden Hypothesen generiert und überprüft. Die Analyse wird als reaktiv (/Duden 10/: Gegenwirkung ausübend od. erstrebend) bezeichnet, weil durch die Ursachensuche der Fehlerzustand behoben werden soll. /Wiegand 04/
Die Wirkungsanalayse stellt eher eine Simulation des Systems dar, die bestimmte Auswirkungen im Systemverhalten vorhersagen möchte. Hier wird von der Ursache auf deren Wirkung geschlossen. /Wiegand 04/

5.2 Aufbau der Station

Abbildung 5.1: *Gesamtdarstellung der Transferstraße „Rotes Wunder“*

Die Transferstraße (fisher Technik, genaue Bezeichnung) besteht (in Laufrichtung des Bandes) aus einem

- Registerlager mit Förderband
- Förderband mit Drehtisch und Rollenbahn,
- Förderband mit Werkzeugmaschine
- Umsetzer mit Sauggreifer und
- Förderband mit zwei Pushern.

Die aus einem Lager stammenden zu transportierenden Teile werden aufgrund eines angebrachten Codes den entsprechenden Stationen zugeordnet. Die Steuerung erfolgt über Labview. Der Steuerung werden dabei über die Sensorik der Anlage (etwa betätigte Schalter) bestimmte Signale übermittelt, auf die diese dann entsprechend ihres Programms reagiert, also etwa einen Motor ansteuert.

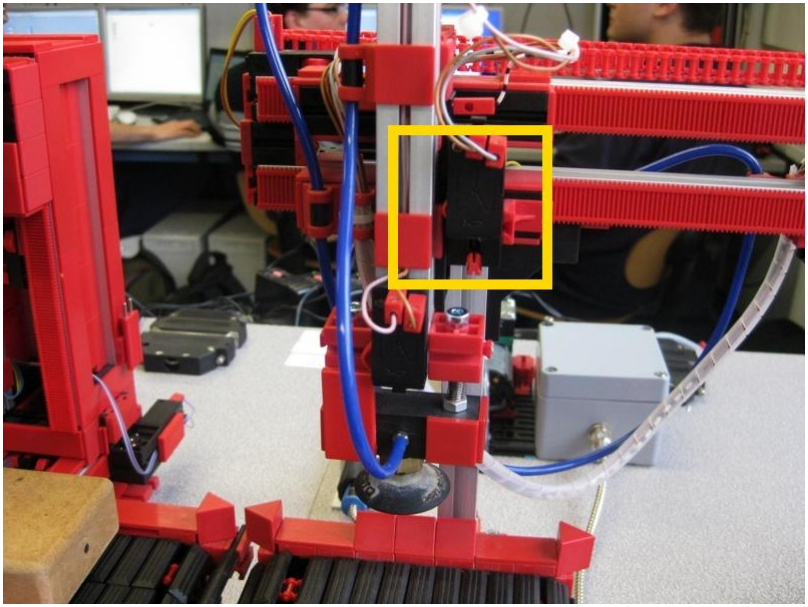


Abbildung 5.2: *Sensorik der Anlage Taster_Bewegung des Umsetzers*

Die Arbeitsweise dieses Modells ist durchaus auch mit dem herkömmlicher Industrieanlagen vergleichbar. Ein vollständiger Bauteilebaum, wie er von den Studierenden erstellt wurde, befindet sich in Anhang A.

5.3 Wissensrepräsentations-Sprache

Die Wissensbasen wurden von den Teilnehmern der Vorlesung in einer externen Repräsentation verfasst, die sich an eine Wissensrepräsentationssprache orientiert, die innerhalb der Vorlesung entwickelt wurde. Bestanteile dieser Sprache sind Konstrukte, die folgende Wissenselemente definieren:

- Bauteil: die einzelnen Bestandteile der Anlage bis zur Austauschtiefe (bei Defekt wird das Teil nicht weiter untersucht, sondern ausgetauscht);
- Bauteilgruppe: besteht aus einzelnen Bauteilen;
- Vor-, Verlaufs- und Nachbedingungen: bilden den Arbeitsvorgang ab, indem sie Bedingungen abbilden, die gelten müssen damit ein Arbeitsvorgang überhaupt startet, damit er korrekt ablaufen kann und die er erzeugt;
- Regeln: bestehen aus Prämissen und Konklusionen;
- Regelgruppen: Zusammenfassungen von Regeln;
- Literale: sind Aussagen oder der Vergleich zwischen Aussagen. Diese können in der Prämisse mit AND, OR und NOT und in der Konklusion mit AND verbunden werden;
- Aussagen – stellen die eigentlichen vom Benutzer zu bearbeitenden Bestandteile dar;
- Koroutinen - diese enthalten die einzelnen Problemlösungsschritte und
- die Gesamtanlage – als oberstes Element, mit dem die Problemlösung beginnt.

Bauteile und Baugruppen werden über die Eigenschaften „besteht-aus“ (Spezialisierung) und „ist-Teil-von“ (Generalisierung) zu einem Bauteilebaum verbunden. Die Elementarobjekte sind dabei Bauteile. Sie stellen die Austauschtiefe dar.

Damit ein Bauteil überhaupt funktionieren kann, müssen bestimmte Startbedingungen erfüllt sein. Diese Vorbedingungen werden dem Bauteil zugeordnet, welches den Prozess ausführt. Der Prozess erzeugt Nachbedingungen, die ihrerseits wieder Vorbedingungen eines anderen Prozesses sind. Während der Prozess läuft müssen Verlaufsbedingungen

gelten, die diesen Prozess „am Leben“ halten (siehe hierzu auch Kapitel 5.5). Die Nachbedingungen eines Prozesses müssen dabei die Vorbedingungen eines anderen Prozesses sein, die Verlaufsbedingungen die Nachbedingungen eines Prozesses, damit dieser verfolgt werden kann.
Sowohl einzelne Regeln, wie Regelgruppen können Bauteilen und Bauteilgruppen zugeordnet werden.
Regelgruppen oder einzelnen Aussagen kann als Strategie die Rückwärtsverkettung zugeordnet werden. Ansonsten werden die Regeln in der Reihenfolge ihres Vorkommens in der Regelgruppe bearbeitet.
Beim Wissenselement Aussage werden Wertebereiche, Frage- und Diagnosetexte, sowie die Anzahl der auszuwählenden Belegungen aus einer Menge angegeben.
Die Koroutinen enthalten die einzelnen Problemlösungsschritte, die vom Inferenzer abgearbeitet werden, sobald das Objekt als erstes Element im Hypothesenfokus (die Liste der zu untersuchenden Objekte) angefasst wird. Folgende Problemlösungsschritte können ausgewählt werden:

- Untersuche Bauteil,
- Untersuche Baugruppe,
- Untersuche Regelgruppe,
- Untersuche Regel,
- Untersuche Bedingungen und
- Hypothesenverfolgung.

Mit dem Inferenzschritt *Hypothesenverfolgung* wird die Koroutine suspendiert. Wird sie erneut aktiv, wird ihre Bearbeitung nach diesem Schritt fortgeführt.

5.4 Problemlösung

An der Problemlösung ist neben den Koroutinen, welche die einzelnen Problemlösungsschritte enthalten auch die Liste, welche zur Verarbeitung der Inferenzschritte verwendet wird, beteiligt. Diese wird hier als *Hypothesenfokus* bezeichnet, weil sie die zu verfolgenden Hypothesen enthält.
Zu Beginn der Problemlösung enthält der Hypothesenfokus das Objekt *Gesamtanlage*.

Die Verarbeitungsschleife hat nun folgenden Aufbau:

1. Nehme das erste Elemente, welches im Hypothesenfokus enthalten ist. Ist kein Element vorhanden beende die Verarbeitung.

2. Ist das Element ein Bauteil oder eine Baugruppe
   Steht der Problemlösungsschritt-Anzeiger auf dem Inferenzschritt „Hypothesenverfolgung“ der entsprechenden Koroutine
   Ist kein nächster Schritt vorhanden
   Entferne das Bauteil oder die Baugruppe aus dem Hypothesenfokus und gehe zu 1.
   Sonst: Gehe zum nächsten Schritt.
   a. Ist der Inferenzschritt *Hypothesenverfolgung*:
   Ordne Bauteil ans Ende des Hypothesenfokus
   Gehe zu 1.
   Sonst: Führe den Problemlösungs-Schritt aus.
   Gehe zu a
3. Ist das Element eine Regel oder eine Regelgruppe verarbeite diese mit Hilfe des Regelinterpreters.

4. gehe zu 1

Die einzelnen Problemlösungsschritte (Inferenzschritte) werden wie folgt verarbeitet:

- Untersuche_Bauteil <Name>
  - Schreibe die Bauteile <Name> an den Anfang des Hypothesenfokus
- Untersuche_Baugruppe <Name>
  - Schreibe die Bauteilegruppe <Name> an den Anfang des Hypothesenfokus
- Untersuche_alle_Bauteile [<Name der Baugruppe>]
  - Schreibe alle im Slot „besteht-aus“ vorhandene Bauteile der Baugruppe an den Anfang des Hypothesenfokus. Wird kein Name genannt so ist die Baugruppe gemeint, deren Koroutine gerade verarbeitet wird.

- Untersuche_alle_Regelgruppen [[von Bauteil | von Baugruppe] <Name >]
  - Schreibe alle im Slot „Regelgruppe“ vorhandenen Regelgruppen des Bauteils oder der Baugruppe an den Anfang des Hypothesenfokus. Wird kein Name genannt so sind die Regelgruppen gemeint, deren Koroutine gerade verarbeitet wird.
- Untersuche_Regelgruppe <Name> [[von Bauteil | von Baugruppe] <Name >]
  - Schreibe die Regelgruppe <Name> des Bauteils oder der Baugruppe an den Anfang des Hypothesenfokus. Wird kein Name genannt so ist die Regelgruppe ein Bestandteil der Liste „Regelgruppen“ des Bauteils/der Baugruppe, deren Koroutine gerade verarbeitet wird.
- Untersuche_alle_Regeln [[von Bauteil | von Baugruppe] <Name >]
  - Schreibe alle im Slot „Regeln“ vorhandenen Regeln des Bauteils oder der Baugruppe an den Anfang des Hypothesenfokus. Wird kein Name genannt so sind die Regeln des Bauteils/der Baugruppe gemeint, deren Koroutine gerade verarbeitet wird.
- Untersuche_Regel <Name> [[von Bauteil | von Baugruppe] <Name >]
  - Schreibe die Regel <Name> des Bauteils oder der Baugruppe an den Anfang des Hypothesenfokus. Wird kein Bauteil/keine Baugruppe genannt so ist die Regel ein Bestandteil der Liste „Regeln“ des Bauteils/der Baugruppe, deren Koroutine gerade verarbeitet wird.
- Untersuche_Bedingungen [[von Bauteil | von Baugruppe] <Name >]
  - Bei jeder unwahren Vorbedingung wird das Bauteil/die Baugruppe weiter untersucht, bei der diese Vorbedingung eine Nachbeding ist. Bei diesem Bauteil/dieser Baugruppe werden die Vorbedingungen und die Verlaufsbedingungen überprüft.
  - Bei jeder unwahren Verlaufsbedingung wird das Bauteil/die Baugruppe weiter untersucht, bei der diese Verlaufsbedingung eine Nachbedingung ist. Bei diesem Bauteil/dieser Baugruppe werden die Vorbedingungen und die Verlaufsbedingungen überprüft.
  - Trifft man während dieser Untersuchung auf Bauteile/Baugruppen bei denen die Vor-und die

Verlaufsbedingungen alle wahr sind, die Nachbedingungen aber fehlerhaft, verursacht das gerade betrachtete Bauteil den Fehler und wird an den Anfang des Hypothesenfokus geschrieben.

- Hypothesenverfolgung
  - Die entsprechende Koroutine wird beendet, das Bauteil an das Ende des Hypothesenfokus geschrieben.

Der Regelinterpreter verarbeitet die Regelgruppen standardgemäß sequentiell, d.h. in der Reihenfolge in der sie in der Regelgruppe vorkommen. Ist die Strategie *Rückwärtsverkettet* bei einer Regelgruppe gesetzt so werden alle Literale der Prämissen entsprechend der in Kapitel 4.6.2 beschriebenen Weise behandelt. Die Rückwärtsverkettung einer Regel bewirkt, dass nur die Literale der Prämisse dieser Regel rückwärtsverkettet werde, die einer Aussage nur die entsprechende Aussage. Eine Vorwärtsverkettung oder die Goalbasierte Verarbeitung ist im System nicht implementiert.
Eine Regel wird nur dann verarbeitet, wenn sie noch nicht verarbeitet wurde. Dazu wird intern die Eigenschaft TRACED bei den entsprechenden Elementen gesetzt. Über diese Eigenschaft wird auch verhindert, dass eine Aussage nochmals erfragt wird und eine Regel nochmals abgearbeitet wird. Bauteile und Baugruppen werden nach vollständiger Abarbeitung ihrer Koroutinen beim Problemlösungsvorgang nicht mehr beachtet. Um den Verarbeitungsvorgang zu beschleunigen, könnte man auch hier eine TRACED Eigenschaft einfügen, die gesetzt wird wenn, das Bauteil zum ersten Mal aus dem Hypothesenfokus gestrichen wird.
Der Regelinterpreter verarbeitet die Prämisse in dem er die einzelnen Bestandteile der Prämisse entsprechend den logischen Regeln verarbeitet und die Prämisse zu verifizieren sucht. Dabei werden die einzelnen Aussagen vom Benutzer erfragt, oder wenn sie schon einmal erfragt wurden (dann wurde TRACED auf True gesetzt) nur deren Wert festgestellt und das entsprechende Literal verarbeitet. Wird die gesamte Prämisse zu wahr inferiert, so feuert die Regel, die Konklusion wird ausgeführt, entsprechende Diagnosen ausgegeben und/oder Inferenzschritte ausgeführt.

5.4.1 Vor-, Verlaufs- und Nachbedingungen

Eine technische Anlage führt einen Arbeitsablauf aus, den man als Prozess verstehen kann. Dieser braucht, um überhaupt starten zu können, bestimmte Voraussetzungen (Vorbedingungen) und zu seinem ordnungsgemäßen Ablauf bestimmte Zustände, die während des gesamten Verlaufs vorhanden sein müssen (Verlaufsbedingungen). Durch die Durchführung des Prozesses werden bestimmte Systemzustände erzeugt, die dann einem nachgeordneten Prozess wieder als Voraussetzungen dienen können (Nachbedingungen).
Wie ein Bauteilebaum sich von der Gesamtanlage in die einzelnen Bauteile herunter brechen lässt, so kann man auch den Hauptprozess immer weiter verfeinern. Dem einzelnen Prozessteil kann dann als Agent ein Bauteil zugeordnet werden. Vor-, Verlaufs- und Nachbedingungen können diesem Bauteil zugeordnet werden. Und damit lässt sich ausdrücken, dass die Baugruppe ihre /das Bauteil seine eigentliche Funktion erst ausführen kann, wenn die Vorbedingungen und Verlaufsbedingungen erfüllt sind und selbst durch ihre/seine Arbeit Nachbedingungen erzeugt.
Bei der Untersuchung dieser Bedingungen schreitet man von den Nachbedingungen zu den Vorbedingungen von einem Agenten zum nächsten. Die Vorbedingungen eines Prozesses sind dabei die Nachbedingungen eines anderen Prozesses, wobei man beim obersten Objekt den eigentlichen Prozess verlässt. Wenn z.B. keine Teile mehr angeliefert werden, kann die Anlage auch ihre Aufgabe nicht mehr erfüllen, der Fehler liegt also nicht mehr im betrachteten Fokus. Er ist für die Diagnose des Fehlerzustands aber dennoch sinnvoll.
Abbildung 5.3 zeigt einen Ausschnitt der Verkettung der Vor-, Verlaufslaufs und Nachbedingungen des „Roten Wunders“.

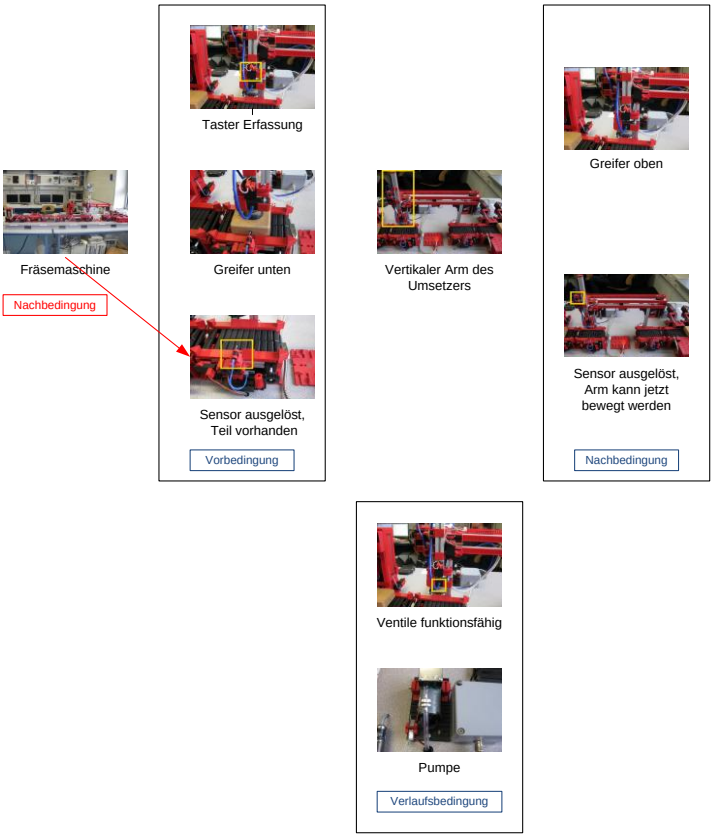


Abbildung 5.3: *Vor-, Verlaufs- und Nachbedingungen des vertikalen Arm*

Bauteile oder –gruppen, deren Vor- oder Verlaufsbedingungen gestört sind, können ihre Aufgaben nicht mehr ordnungsgemäß durchführen. Sie können also nicht die eigentliche Ursache des Fehlers darstellen. Sind Vor- oder Verlaufs-Bedingungen gestört verweisen diese entweder direkt auf die Prozesse, deren Nachbedingungen sie darstellen oder, falls diese im

Problembereich nicht mehr dargestellt werden, stellen selbst die gesuchte Diagnose dar. Agenten, die aber trotz bestehender Vor- und Verlaufsbedingungen nicht im Stande sind die korrekten Nachbedingungen zu erzeugen, sind selbst gestört und müssen untersucht werden. (siehe hierzu auch /Gens 90/)

5.5 Beispielwissensbasen

In diesem Kapitel möchte ich kurze einige Beispiel für die mit der oben beschriebenen WRS entstandenen Wissensbasen geben. Dabei werden jeweils

Ausschnitte dargestellt, um dem Leser einen Eindruck zu vermitteln.

```
Gesamtanlage:
        Name => Rotes_Wunder,
        besteht_aus =>      {       Foerderband_mit_2_Pushern,
                                    Umsetzer_mit_Sauggreifer,
                                    Gesamtanlage_Werkzeugmaschine,
                                    Foerderband_Drehtisch_Rollenbahn,
                                    Registerlager_mit_Foerderband},

Regeln =>           { Wurzelregel },
Regelgruppen =>     {Leitsymptome },

Koroutine =>        {               Untersuche_Regelgruppe Leitsymptome,
                                    Hypothesenverfolgung};
Regelgruppe:
        Name => Leitsymptome,
        Regeln =>           {Wurzelregel,
                            Untersuche_Foerderband_mit_2_Pushern_OK, …};
Regel:
        Name => Wurzelregel,
        Praemisse => {
                                    Foerderband_mit_2_Pushern_OK AND
                                    Umsetzer_mit_Sauggreifer_OK AND
                                    Gesamtanlage_Werkzeugmaschine_OK AND
                                    Foerderband_Drehtisch_Rollenbahn_OK ND
                                    Registerlager_mit_Foerderband_OK
                                    },
        Konklusion => {
                                    Alles_OK };
```

Die Problemlösung beginnt hier mit der Untersuchung des Objekts „Gesamtanlage“. Zunächst wird eine Wurzelregel (Leitsymptome) untersucht, die zunächst auf Störungen der einzelnen Baugruppen abprüft. Führt diese Regel zu keinem Ergebnis, findet das System keine Störungen und wird beendet. Wird eine Störung gefunden, wird bei der Untersuchung der einzelnen Bestandteile fortgefahren (hier zur Verkürzung nach der Regel Untersuche_Foerderband_mit_2_Pushern_OK mit … abgekürzt).
Innerhalb dieser Regeln kann dann die Untersuchung von Bauteilen oder Baugruppen veranlasst werden (Inferenzschritt in der Konklusion).

```
Baugruppe:
        Name => Umsetzer_mit_Sauggreifer ,
        besteht_aus => {
                        Foerderband_Links,
                        Foerderband_Rechts,
                        Zwischenlager,
                        Horizontaler_Arm,
                        Vertikaler_Arm,
                        Pumpe
                        },
        ist_teil_von =>     { Rotes_Wunder },
        Vorbedingungen => {Teil_fertiggestellt_durch_Fräsemaschine}
        Verlaufsbedingungen     =>     {Stromversorgung_OK     AND
Steuerung_ansprechbar}
        Nachbedingungen => {Induktiver Nährungsschalter rechts = „ausgelöst“}

        Regeln => {},
        Regelgruppen =>   {},
        Koroutine =>      {
                        Untersuche_Bedingungen,
                         Hypothesenverfolgung,
                        Untersuche_Baugruppe Foerderband_Links,
                        Untersuche_Baugruppe Foerderband_Rechts,
                        Untersuche_Baugruppe Horizontaler_Arm,
                        Untersuche_Baugruppe Vertikaler_Arm,
                        Untersuche_Baugruppe Zwischenlager,
                        Untersuche_Baugruppe Pumpe,
                        Hypothesenverfolgung
                        },
        Medien =>         {Bild
"Gesamtanlage\Umsetzer_mit_Sauggreifer\Umsetzer_mit_Sauggreifer.jpg"};
```

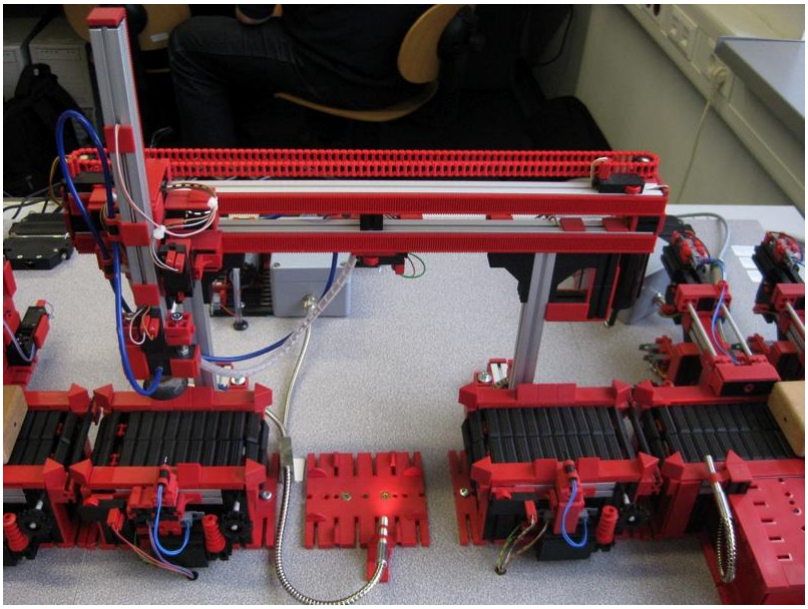


Abbildung 5.4: *Umsetzer mit Sauggreifer*

Das in der Abbildung links ankommende Teil muss durch die Fräsmaschine korrekt verarbeitet worden sein. Ist dies nicht der Fall, kann ein möglicher Fehler nicht dem Umsetzer zugeordnet werden. Während der Umsetzer arbeitet, braucht er beständig Strom und Zugriff auf die Steuerung. Nachdem er seine Arbeit durchgeführt hat, ist das zu bearbeitende Teil nach rechts transportiert worden und hat während dieses Vorgangs den Sensor Induktiver-Näherungsschalter rechts angesprochen (Abbildung 5.5).

Durch die Untersuchung der Bedingungen können andere Baugruppen und –teile als mögliche Fehlerquelle ermittelt worden sein, weshalb es sinnvoll ist die entsprechende Koroutien mit dem Inferenzschritt *Hypothesenverfolgung* zu verlassen. Erst wenn die Untersuchung dieser Fehlerquellen zu keinem Erfolg geführt hat, kann die Koroutine weiterverarbeitet werden.

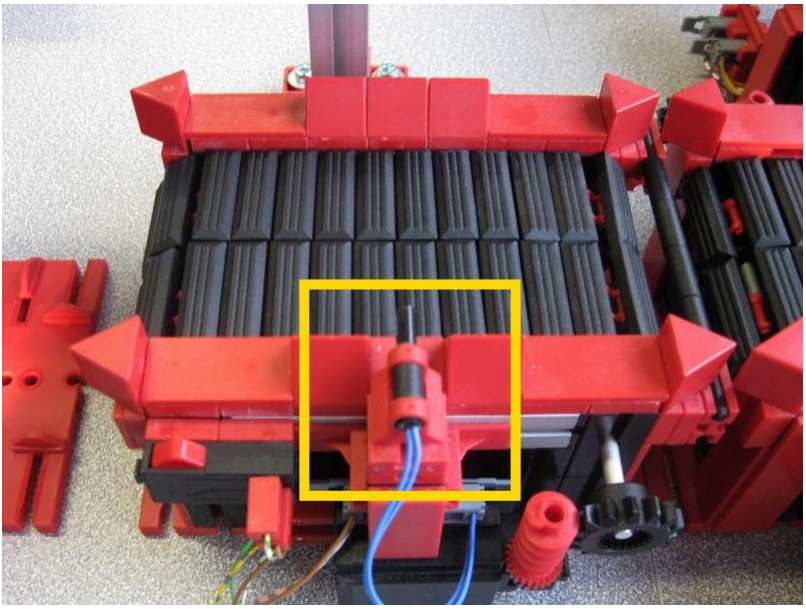


Abbildung 5.5: *Induktiver Näherungsschalter rechts*

```
Bauteil:
        Name =>             Motor_Rechts,
        Asset_ID =>         7M4,
        ist_Teil_von =>     { Foerderband_Rechts },
        Regeln =>           { Motor_OK,  Strom_nicht_OK, Motor_Stillstand },
        Regelgruppen =>     {Fehler in der Motoreinheit },
        Koroutine =>        {Untersuche_Regelgruppe Fehler in der Motoreinheit,
                             Hypothesenverfolgung},
        Medien =>           {Bild
"Gesamtanlage\Umsetzer_mit_Sauggreifer\Foerderband_Rechts\Motor_Rechts.jpg"};
Regel:
        Name =>             Motor_ OK,
        Praemisse =>        {
                            (NOT Strom _vorhanden AND
                            Motor_Zustand = „lauffähig“) OR
                            (Strom _vorhanden AND
                            Motor _Zustand= „steht still“)}
        Konklusion => { Textausgabe "Der Motoreinheit am Förderband rechts ist
                                     defekt!" };
Regel:
        Name =>             Strom_nicht_OK,
        Praemisse =>        { NOT Strom _vorhanden},
        Konklusion =>       { Diagnoseausgabe Strom_vorhanden };
Regel:
        Name =>             Motor_ Stillstand,
        Praemisse =>        Motor_Zustand = „steht still“ },
        Konklusion =>       { Textausgabe "Der Motor ist kaputt und muss
                                           ausgewechselt werden!" };
Aussage:
        Name =>             Strom_vorhanden,
        Wertebereich =>     bool,
        Fragetext =>        { "liegt Strom am Motor an? (ja / nein)"};
        Diagnosetext => { "Die Stromversorgung ist nicht vorhanden!"};
Aussage:
        Name =>             Motor_Zustand,
        Wertebereich =>     Liste („lauffähig“, „steht still“)
        Fragetext =>        {"Wie verhält sich der Motor, falls der Strom vorhanden
                              ist:"};
```

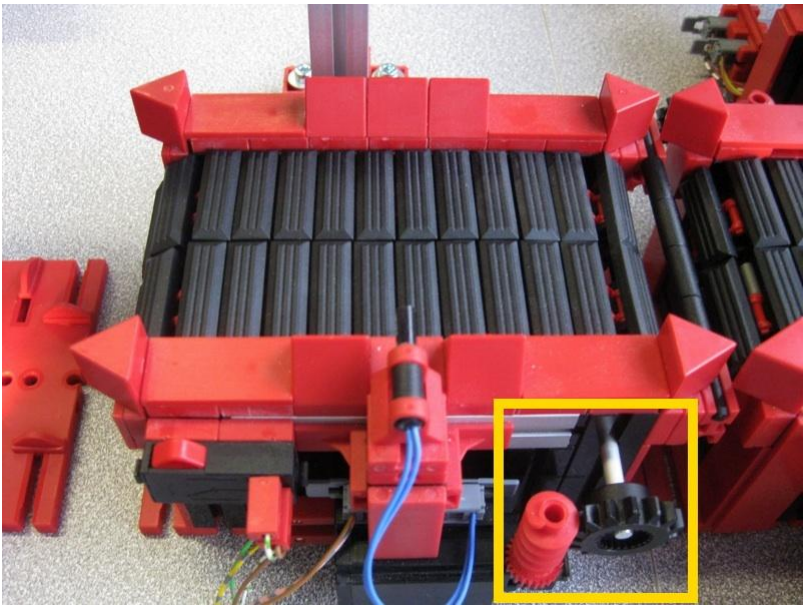


Abbildung 5.6: *Motor_rechts*

Zu dem hier angegebenen Beispiel für die Regelverarbeitung möchte ich folgende Anmerkungen machen:

- Die Regeln und Regelgruppen sollten immer den Bauteilen zugeordnet werden, deren Fehler sie auch behandeln;
- Asset_ID stellt die Nummer im Schaltplan dar, so dass der Nutzer auch dort die Ansteuerung durch die Steuerung ergründen kann;
- Bei der Aussage, die eine Liste als Wertebereich enthält, muss die Dialogkomponente diese Liste entsprechend verwalten. Prinzipiell kann aus einer Liste mehr als ein Element ausgewählt werden. Will man zum Beispiel mehr als ein Objekt aus einer Liste wählen, so muss die Mehrfachauswahl auf true gesetzt werden. Standardmäßig wird ein Wert ausgewählt;
- Die Ausgabe von Texten in der Konklusion kann verschiedenartig bewerkstelligt werden. Über eine direkte Textausgabe wird der angegebene Text ausgegeben, über das Codewort Diagnoseausgabe

wird der Diagnosetext ausgegeben, welcher der Aussage zugewiesen wurde.

6 Knowledge Engineering

Über das Knowledge Engineering wird eine Wissensbasis für ein Expertensystem erstellt. In der Literatur (z.B. /Universität Würzburg 03//Wagner 93//Bachelier 04/) wird dieser Begriff aber sehr leicht als die Erstellung eines gesamten Expertensystems zusammengefasst, so als ob der Knowledge Engineer Programmierer, Experte für Repräsentation und Modellierung und Ingenieur, Informatiker und Psychologe sein muss.

Die Erstellung eines XPS macht aber ein ganzes Team von Spezialisten notwendig. So gibt es vernünftigerweise neben dem Knowledge Engineer den Programmierer, den Designspezialisten und den Fachmann für das Abbildungsgebiet. Die Erstellung eines gesamten Systems umfasst nach meiner Erfahrung einen Arbeitsaufwand von acht bis zehn Mannjahren. Hier mit ein oder zwei Mitarbeitern zu versuchen etwas Sinnvolles auf die Beine zu stellen ist illusorisch.

Abbildung 6.1 zeigt an welchem Punkt einer XPS Erstellung das Knowledge Engineering sich von der Systementwicklung trennen sollte. Um alle Aspekte einer Entwicklung zu berücksichtigen ist es aber angebracht, dass die Machbarkeitsstudie vom gesamten Team durchgeführt wird, die Spezialisierung auf die verschiedenen Arbeiten erfolgt erst wenn diese erfolgreich durchgeführt wurde und die Vorbereitung mit einer Grobplanung abgeschlossen wurden.

Innerhalb der Vorbereitung zur Erstellung eines XPS muss zunächst eine Machbarkeitsstudie (Vorbetrachtungen, Problemanalyse und Umfeld-Analyse) durchgeführt werden, in der untersucht werden sollte, ob die Voraussetzung zur erfolgreichen Entwicklung und zum Einsatz eines XPS überhaupt vorhanden sind.

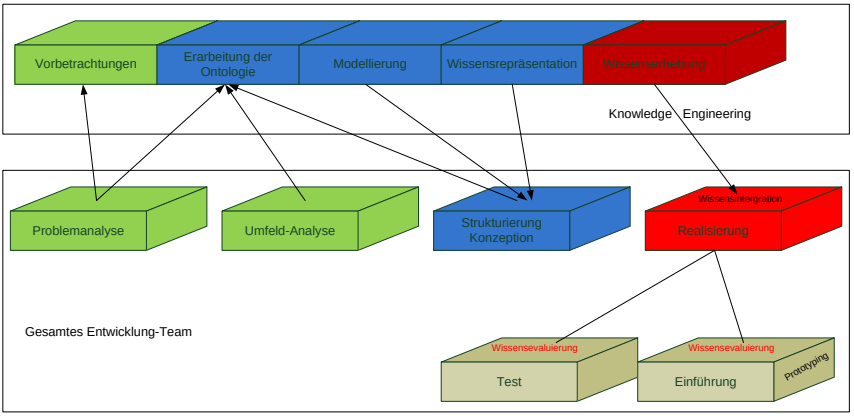


Abbildung 6.1: *Einbettung des KE in die Entwicklung eines XPS*

Innerhalb dieser Machbarkeitsstudie muss abgeklärt werden:

- Welche Ziele verfolgt der Auftraggeber durch den Einsatz des XPS und sind diese realistisch.
  - Die Erstellung eines XPS kostet meist sehr viel Geld und es fallen auch noch Kosten zur Wartung des Systems an. Die Erstellung der von mir in der Industrie erstellten Systeme hatten pro Stück etwa 1 Million DM an Kosten verursacht. Und diese Kosten waren dabei nur so gering, weil wir ein sehr erfahrenes Team von 4 bis zeitweise 7 Mitarbeitern einsetzten konnten. Was ein XPS kann und was nicht ist den Systementwicklern klar und muss so auch an die Kunden weitergegeben werden. Ein XPS kann zum Beispiel nie einen Experten vollständig ersetzen, es kann aber die Qualität des Problemlösungsvorgangs verbessern.
  - Gibt es eventuelle Expertensystem Shells oder Tools, die eingesetzt werden können oder muss ein vollständiges System entwickelt werden, um die Aufgabe zufriedenstellend zu lösen.
- Kann der Erstellungsvorgang überhaupt von dem Auftraggeber unterstützt werden

    - Es macht keinen Sinn, wenn es überhaupt keinen Experten vor Ort gibt, der sein Wissen einbringt. Neben dieser sofort ins Auge fallenden Bedingung ist es aber auch unerlässlich, dass das gesamte Management an der Erstellung eines Systems interessiert ist. Merkt man erst innerhalb der Erstellung des Systems, dass einer der Manager dagegen eingestellt ist, produziert man schnell Schrankware.
- Wie sieht der Einsatzort des Systems aus
    - Dabei muss genau geklärt werden welche Hard- und Software eingesetzt werden kann und wie die Benutzer das System verwenden wollen. Um dies abzuklären ist es unumgänglich eine Vorort-Besichtigung zu machen, in der man auch mit Experten und Managern verschiedener Stufen über das System redet.
- Erst wenn die bisher beschrieben Punkte zur Zufriedenheit sowohl der Entwickler wie des Kunden geklärt werden konnten sollte man
    - ein entsprechendes Entwicklungsteam zusammenstellen, dem zumindest 2 bis 3 erfahrende Programmierer, ein Knowledge Engineer und ein Mitarbeiter, der sich schnell in die abzubildende Domäne einarbeiten kann, umfasst. Nach meiner Erfahrung ist das Knowledge Engineering selbst wieder eine Teamarbeit. Wenn man als Domäne eine Produktionsanlage hat, ist es daher ratsam einen Maschinenbauer mit im Team zu haben. Sollte sich der Spezialist der Wissensrepräsentation erst immer in eine Domäne einarbeiten müssen, kommt es zu vielen Reibungsverlusten. Außerdem ist die Akzeptanz vor Ort größer, wenn ein „Mann vom Fach“ mit auftaucht und der Spezialist der Repräsentation (meist ein Informatiker) nicht alleine die Welt verstehen soll.
    - Von diesem Team wird dann zum Abschluss der Vorarbeiten eine Pflichtenheft erstellt, das zum Beispiel die mit dem Kunden erstellten Anforderungen, wie zum Beispiel die Arbeitsweise und Kurzbeschreibung des Systems, und das organisatorische Umfeld, sowie die Ziel und Meilensteine, die erreicht werden sollten, enthalten sollte. Meine eigene Erfahrungen werden hier auch durch die Literatur gestützt (siehe zum Beispiel /Computerwoche 87/). Bereits hier sollten wichtige Begriffe, die firmenspezifisch sind grob abgeklärt werden. Die Abschlussdokumentation muss an dieser Stelle dann eine vom

KE zusammen mit den Experten erstellten Ontologie enthalten. Diese ist vor allem für die Abnahme des Systems wichtig.

Die Abschlussdokumentation, die dem Kunden mit dem System übergeben wird und die Grundlage der Abnahme darstellt, sollte neben der oben bereits dargestellten Begriffsabklärung (Begriffslexikon, Ontologie) noch:

- Typische Problemlösungssituationen mit Fragen vom und an das System,
- Teilprobleme und
- eine Beispielsammlung (Probleme und Lösungen) /Computerwoche 87/

enthalten.

Zur Angabe gehört aber auch ein Benutzerhandbuch, das den Benutzer detailliert ins System einführt. Die Erstellung dieses Dokuments ist aber am besten durch entsprechende Spezialisten, etwa der technischen Dokumentation, durchzuführen. Wer wirklich nachrechnet, wie teuer es ist eine Dokumentation von Ingenieuren durchführen zu lassen, die dann doch nur suboptimale Ergebnisse liefern können, wird sich meiner Meinung leicht anschließen. Leider ist diese Kunde aber noch nicht zu allen Chefs durchgedrungen. Ein weiterer positiver Aspekt ist es, dass die Entwickler ihr System durch die Gespräche mit den Dokumentationsspezialisten nochmals kurz überprüfen und möglicherweise noch kleine Schwächen feststellen können.
In der Literatur wird die Entwicklung eines XPS grob in die Bereiche Vorbereitung, Modellierung und Validierung eingeteilt (siehe zum Beispiel /Uni Würzburg 04/). Ich habe diese Bereiche nach meinen Erfahrungen etwas angepasst und beleuchte hier die Vorarbeiten nur ganz kurz. Der Vorgang des KE geht meiner Meinung nach der gleichen Weise vor, gleichgültig ob eine XPS-Shell verwendet wird oder ein vollständig neues System gebaut werden muss.

Ausgehend von diesen Vorüberlegungen teile ich den eigentlichen Prozess des KE in die Bereiche:

- Wissenserhebung
- Wissensmodellierung (mit der Erarbeitung der Ontologie und der Wissensrepräsentation)
- Wissensintegration (Teil der Realisierung) und

- Wissensevaluation (Teil des Systemtests und der Einführung)

ein, die in den folgenden Kapiteln ausführlich beschrieben werden sollen.
Die Wissenserhebung stellt dabei die Verwendung und Auswahl von Methoden dar, mit denen Wissen von verschiedenen Quellen erworben und repräsentiert werden kann. Sie beginnt streng genommen in der ersten Sekunde in der man sich mit dem System befasst und endet, wenn sein Betrieb eingestellt wird, weil man selbst noch in der Evaluierung und Wartung neues Wissen erheben muss.
Die Modellierung versucht ein verständliches Modell des Problembereichs zu erarbeiten und wird von dem durch die Erhebung aufgenommenen Wissen gespeist, beginnt aber auch bereits mit der ersten Befassung mit dem System.
Ist die Modellierung bis zu einem gewissen Stand gekommen kann die Wissensbasis aufgebaut und schrittweise getestet werden.
Nach dem die Wissensbasis einen entsprechenden Reifegrad erreicht hat ist es sinnvoll das System schrittweise in die Firma einzuführen und die Endbenutzer mit ihm vertraut zu machen. Die Evaluierung der Wissensbasis beginnt schon mit diesem Schritt und endet während der Lebensdauer des Expertensystems streng genommen nicht mehr. Wissensbasen müssen beständig gewartet und angepasst werden.
Ein System kann nur allmählich überprüft werden, weil es sich in der Realität als zu komplex und groß erweist, um vom zukünftigen Anwender schnell verstanden werden zu können. Umfangreiche Schulungen und Testphasen sind daher vorzusehen.

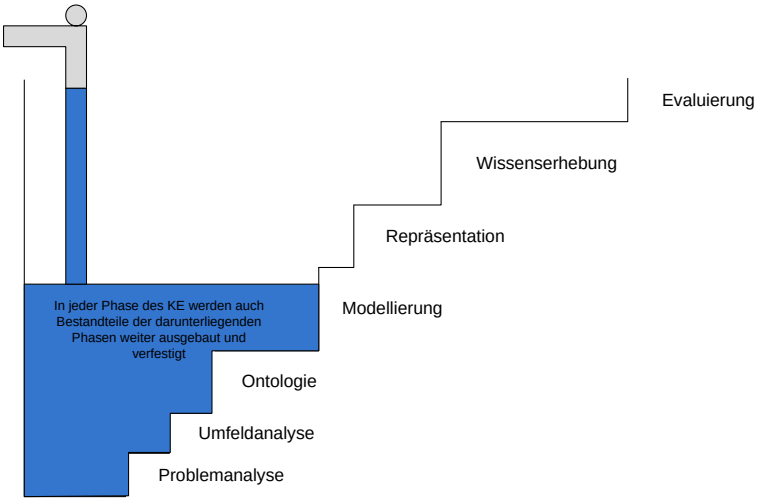


Abbildung 6.2: *Phasenmodell (Badewannenmodell) des KE-Prozesses*

6.1 Wissenserhebung

Die Wissenserhebung (Wissenserwerb, Akquisition) stellt den zentralen Punkt bei der Erstellung eines XPS dar, denn durch diesen Prozess wird das Wissen des Problembereichs vom eigentlichen Experten vor Ort aufgenommen. Sie stellt damit den ersten Schritt zur Erstellung der Wissensbasis dar, ist aber von der Umsetzung des Wissens in eine Präsentation, von mir als Wissensmodellierung bezeichnet abzugrenzen.

Im Rahmen der Wissenserhebung werden verschiedene Techniken verwendet, um das Wissen von verschiedenen Quellen aufzunehmen. Diese Quellen sind zum einen ein oder mehrere Experten vor Ort, aber auch die zur Verfügung stehenden Dokumente oder Sekundärliteratur.

In den Prozess sind die Konowledge Engineers (Wissemsingenieure) und die Experten involviert, wobei hier davon ausgegangen werden soll, dass die Abstimmung mit dem Management bereits im Vorfeld abgeschlossen sein soll.

Zur Durchführung der Akquisition werden verschiedene Techniken verwendet, von denen die hier näher erläuterten Interviewtechniken, die Erstellung von Musterlösungen und das Festlegen des

Problemlösungsvorgangs (Lösungsstrategie) die für mich bedeutendsten sind. /Cawsey 03//Brünner 02/

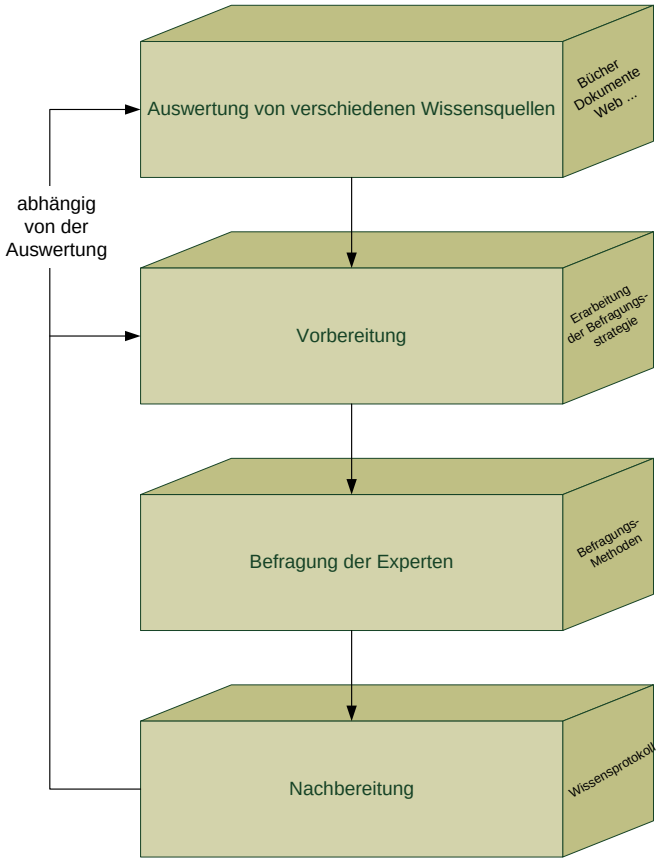


Abbildung 6.3: *Der Prozess der Wissenserhebung*

6.1.1 Der Wissensingenieur

Der Wissensingenieur hat die Aufgabe das Wissen eines Problembereichs zu erarbeiten und es in einer geeigneten Repräsentationsform so darzustellen, dass das XPS es verarbeiten kann. Dabei muss der KE selbst in hohem Maße lernfähig sein, denn er wird wahrscheinlich nicht von Anfang an das betrachtete Gebiet in ausreichender Form überblicken können, um die Informationen des Experten richtig einordnen zu können. Darüber hinaus muss er aber auch über ein hohes Maß an Einfühlungsvermögen verfügen, wenn er den Experten nicht verschrecken will. /Cawsey 03// Wachsmuth 94/ /Wagner 93/

Auf welche Probleme man in diesem Zusammenhang in der Praxis vor Ort stößt, habe ich in meinem Buch zur Repräsentation kurz dargestellt /Hartmann 09/.

Es scheint mir hier noch einmal angebracht zu sein, darauf hinzuweisen, dass man ähnlich wie beim Tauchen, beim Knowledge Engineering niemals allein sein sollte, allzu leicht begibt man sich sonst in Gefahr. Es hat sich in meiner Praxis bewährt, dass ein Knowledge Engineering – Team gebildet wurde, dass aus einem Fachmann für das abzubildende Gebiet (in unserem Fall einem Maschinenbauingenieur) und einem Fachmann für die Wissensrepräsentation (einem Informatiker) gebildet wurde.

Die Einrichtung eines Teams erleichtert nicht nur die eigentliche Befragung der Experten vor Ort, die in unserem Fall schon aus Kostengründen eine ganze Schicht dauerte, sondern auch die Vor- und Nachbereitung der Befragung. Auch die Einarbeitung in ein Wissensgebiet geht für den hier nicht versierten Informatiker wesentlich schneller, wenn er sich nicht mit allen Trivialitäten an den Experten wenden muss, sondern einen Gesprächspartner in der eigenen Firma besitzt. Darüber hinaus ist die Befragung eines Experten auch davon abhängig, wie groß das Vertrauen des Experten in die Kompetenz des KE ist. Und diese wird dadurch gestärkt, dass einer der Beteiligten des Befragungsprozesses „die gleiche Sprache“ wie der Experte spricht.

Werden mehrere Experten in die Wissenserhebung einbezogen, muss der Knowledge Engineer deren Wissen in der Wissensbasis so zusammenfassen, dass für den Nutzer ein einheitliches Bild entsteht. Nutzer und Experte sind meist nicht die gleichen Personen und der Nutzer versteht nicht immer warum sich ihm an einer bestimmten Stelle ein besonderer Problemlösungsvorgang zeigt. Dieser Vorgang der Vereinheitlichung kann für den KE sehr schwierig sein, zumal sich auch die jeweiligen Experten immer wiederfinden sollen.

Die in der Literatur anzutreffende Meinung, dass der KE eine „Codierung" des Wissen in eine maschinennahe Sprache vornehmen müsste (z.B. /Wagner 93/), geht meiner Meinung nach weit an der Realität vorbei. Wissensingenieure sind keine eierlegende Wollmilchsäue sondern arbeiten bei der Entwicklung von Expertensystemen mit Programmierern zusammen, die eine interne Repräsentation des Wissens in einer Programmiersprache erarbeiten. Bei der Entwicklung eines Expertensystems in der Praxis wird zunächst eine externe Wissensrepräsentationssprache entwickelt, die durch ein entsprechendes Programmmodul in eine interne Repräsentationssprache umgesetzt wird. Der Wissensingenieur arbeitet sinnvollerweise auf diese, auch vom Experten lesbaren, externen Sprache. Er codiert nicht, dazu hätte er überhaupt keine Zeit. Allein die Entwicklung einer realen Wissensbasis kostet ein bis zwei Mannjahre. Meiner Meinung nach können solche Meinungen nur von Wissenschaftlern vorgenommen worden sein, die sich an kleinen XPS versuchten und selbst Programmierer und KE sein mussten.
Wird bei der Erstellung einer Wissensbasis eine Shell oder eine Tool verwendet, so muss sich der Wissensingenieur an die hier festgelegte Repräsentationsformen halten. Dem zeitlichen Vorteil, sofort ein ablauffähiges System zu haben, steht hier der Nachteil gegenüber das Problem anpassen zu müssen. Nicht jeder Shell ist für die Lösung jedes Problems sinnvoll einsetzbar.

6.1.2 Der Experte

Die Auswahl der Experten /wiki 10b/, deren Wissen man im XPS abbilden möchte, ist das wohl diffizilste Problem, dass innerhalb der Realisierung gelöst werden muss. In der Realität kann man den Experten nicht einfach aus dem Arbeitsprozess nehmen und befragen. Gerade weil er Experte ist, wird er in seinem normalen Arbeitsfeld gebraucht.
Ein einfaches Beispiel aus meiner Praxis als Knowledge Engineer mag hier ein Schlaglicht auf die Problematik werfen. Bei der Entwicklung eines Expertensystems für VW in Braunschweig hatten wir die Sitzungen mit unserem Experten so gelegt, dass wir mit dem Zug morgens anreisen konnten und abends auch wieder nach Berlin zurückkamen. Wir waren etwas überrascht, dass wir meist einen übelgelaunten Experten vor uns sahen und gingen dem auf den Grund.
Der Experte begann seine Schicht um 6:00 Uhr morgens. Wenn er um 10:00 eine Sitzung mit uns hatte, musste er morgens die Anlage erst übernehmen, diese dann kurz vor der Sitzung an einen andren Mitarbeiter übergeben und

nach der Sitzung wieder übernehmen, um sie dann am Ende seiner Arbeit wieder zu übergeben.
In Kenntnis dieses Vorgangs verlegten wird den Anfang unserer Sitzungen auf 6:00 Uhr morgens und führten diese, mit entsprechenden Pausen, auch bis zum Schichtende durch. Der Experte konnte dann eine ganze Schicht tauschen und musste erst gar nicht in den Blaumann umsteigen.
Ob bei der Wissenserhebung ein oder mehrere Experten eingesetzt werden sollen – wenn es diese überhaupt gibt - wird in der Literatur eher kontrovers diskutiert /Wagner 93/. Der Einsatz eines Experten lässt die Wissensbasis in sich geschlossener erscheinen. Mehrere Experten werden die Wissensbasis möglicherweise aufgrund ihres, in der Summe größeren Wissens, vollständiger machen können. Die Wissensbasis wird dann aber auch in sich weniger geschlossen sein, weil jeder Experte eine etwa andere Methodik hat Wissen anzuwenden und zu strukturieren. Eine solche Wissensbasis kann schon nach kurzer Zeit nicht mehr wartbar sein, weil der Wissensingenieur eine Vereinheitlichung vornehmen wird und im Nachhinein vielleicht nicht mehr feststellbar ist, welcher Experte welches Wissen eingebracht hat.
Die in der Literatur ebenfalls geäußerte Vorstellung, dass umso mehr Experten benötigt würden je grösser die Wissensbasis sein sollte, auf die Wagner ebenfalls hinweist, halte ich allerdings generell für falsch. Es geht einer Wissensbasis dann eher so wie dem Brei mit den vielen Köchen. Allenfalls zur Überprüfung schon realisierter Wissensbasen können verschiedene Experten hilfreich sein /Wagner 93/. Dann kann die Geschlossenheit der Wissensbasis um weiteres Wissen ergänzt werden, sie bleibt aber in ihrer Struktur einheitlich. Ergänzungen anderer Experten sollten in der Wissensbasis auch immer als solche kenntlich gemacht werden.
Jede Kommunikation zwischen Wissensingenieur und Experten muss außerdem auch unter einem psychologischen Aspekt betrachtet werden. Innerhalb der Gespräche müssen sich die beiden Parteien auch aneinander gewöhnen und ein Vertrauensverhältnis aufbauen können. Achtung vor der Arbeit des Experten erscheint mir hier notwendig. Allzu oft wird dies im akademischen Umfeld aber vergessen. Je mehr verschiedene Kommunikationspartner der Wissensingenieur hat, umso schwieriger wird es für ihn allen gerecht zu werden.

6.1.3 Erhebungsmethoden

a) Vorbereitung
Im Vorfeld der eigentlichen Erhebung müssen zunächst organisatorische Probleme gelöst werden(siehe hierzu auch /Wagner 93/):

- Wo soll die Erhebung stattfinden?
- Welche Zeitdauer und welche zeitliche Abstand zwischen den Erhebungen wird eingeplant?
- Was soll das Ergebnis einer Erhebung sein?
- Welche Phasen soll die Gesamtakquisition haben?

Wenn man davon ausgeht, dass KE und Experte nicht zur gleichen Firma gehören, so wird durch den Ort der Erhebung festgelegt, wer Reisen muss. Findet die Erhebung am Einsatzort statt, können eventuell auftretende Probleme direkt „im Feld" geklärt werden. Findet die Erhebung beim Knowledge Engineer statt, ist der Experte aus einem eigenen Arbeitsumfeld herausgelöst und die Erhebung kann eventuell ungestörter erfolgen. In der Firma kann der Experte jederzeit zu einem speziellen Problem gerufen werden.
Um eine Erhebung durchzuführen und vorzubereiten, bedarf es von Seiten des KE einer Vor- und Nachbereitung. So müssen Befragungen im Vorfeld gut vorbereitet werden, weil man nicht nur ein lockeres Gespräch mit dem Experten führen will, sondern einen bestimmten Bereich der Wissensbasis erstellen möchte. Jede Woche Expertengespräche zu führen ist nach meiner Erfahrung nicht sinnvoll, weil in so kurzer Zeit die Nachbereitung der letzten Sitzung nicht durchgeführt werden kann. Die Zeitdauer der Befragung richtet sich nach dem Experten selbst und der Kondition des KE. In meiner Praxis hat es sich bewährt, dass ein Team von KEs die Erhebung durchführte und diese sich im Laufe einer Besprechung immer wieder abwechselten. Aus oben schon kurz erwähnten Gründen dauerte eine Befragung in unserem Falle solange wie die Schicht des Experten.
Falls keine Shell verwendet werden kann, wird zunächst die Erstellung einer entsprechenden Wissensrepräsentations-Sprache stehen, in der die Wissensbasis abgebildet wird. Wird eine Shell verwendet, muss das Problem auf die Wissensrepräsentations-Sprache dieser Shell angepasst werden.
/Wachsmuth 94/ bezeichnet die aus den eigentlichen Besprechungen entstehenden Dokumente als Wissensprotokolle. Diese stellen die auch für die Dokumentation wichtigen Ergebnisprotokolle dar. Die externe Repräsentation

wird in dieser Phase mit dem Experten feinabgestimmt, so dass sie für ihn auch lesbar wird (siehe hierzu Kapitel 5). Später werden die Ergebnisse direkt in die Wissensbasis überführt und dem Experten das System zugänglich gemacht bzw. wird das System dem Experten zyklisch vorgeführt.
Die Erstellung einer Wissensbasis erfolgt in verschiedenen Phasen, die im Vorab festgelegt werden sollten. So wird man zum Beispiel in der technischen Diagnostik zunächst eine Grobstruktur der zu untersuchenden technischen Anlage vornehmen und diese dann im Laufe des Knowledge Engineerings verfeinern. Dabei wird der Wissensingenieur sinnvollerweise von Stücklisten ausgehen, die er zunächst einfügt, und anhand dieser Grobstruktur den Experten dann näher befragen. Im Allgemeinen muss der KE sich ein Gerüst schaffen, dass er nach und nach ausbaut, um die Komplexität in den Griff zu bekommen.

b) Methoden

Für die eigentliche Wissenserhebung werden in der Literatur eine große Anzahl von Methoden genannt, die man in direkte und indirekte Methoden unterteilen kann/Eisfeld 01/. Als direkte Methoden werden die verstanden, bei denen der Experte und der Wissensingenieure direkt zusammenarbeiten, wie etwa Interviews und Beobachtung. Innerhalb der indirekten erarbeitet sich der KE das Wissen nicht direkt vom Experten. Hier wird es zum Beispiel indirekt aus den bereits vorliegenden Ergebnissen neues Wissen ableiten.

Zu den direkten Methoden zählen:

- Interviews,
- Protokollanalyse (auch als Laut-Denken-Protokoll bezeichnet),
- Feldbeobachtung,
- Gruppendiskussion,
- Brainstorming,
- Review (Simulation).

Die indirekten Methoden lassen sich hingegen in:

- Dokumentenanalyse (Inhaltsanalyse und Sekundäranalyse)
- Introspektion,
- schriftliche Befragung,
- Delphi-Methode
- Dialog mit Klienten (des Experten),

- Soziometrie

unterteilen.

Von diesen Methoden ist die automatische Wissenserhebung zu unterscheiden, in der maschinelles Lernen eingesetzt wird, um aus Fallbeispielen selbst Wissen zu extrahieren. /Bachelier 04//Habel 85/
Die wohl wichtigste Erhebungsmethode stellt das Interview dar, das sich nochmals in verschiedene Arten unterteilen lässt. So werden in der Literatur (z.B. /Universität Würzburg 03//Wagner 93//Eisfeld 01//Porr 02/) folgende Interviewarten genannt:

- das *unstrukturierte Interview*
- *semistrukturierte Interviews:* vorgegebene Themengebiete
- *strukturierte Interviews:* genau vorgegebene Fragen
- *"teachback " Interview:* Wissensingenieur demonstriert sein Verständnis durch Umschreiben oder Lösen eines Problems
- *tutorielles Interview*: Experte hält Vorlesung (siehe hierzu auch /Lenz 07/- narratives Interview; Narration, die: (veraltet) Erzählung, Bericht /Duden 10/)
- Intensivinterview (Fallbesprechung).

Beim unstrukturierten Interview steht im Voraus kein festes Thema fest. Der Experten wird einfach zum Reden ermuntert. Der KE führt aufgrund der Erzählung des Experten in die verschiedenen angesprochenen Gebiete, ohne aber allzu sehr in die Tiefe zu gehen. Unstrukturierte Interviews werden zu Beginn der Befragungen und als Abschluss einer Verfeinerungsphase durchgeführt. Hier dienen sie auch dazu noch einmal den Bereich zusammenzufassen.
Strukturierte Interviews sind vom Wissensingenieur genau vorbereitet. Dabei kann auch mit schon vorbereiteten Fragen und Antworten gearbeitet werden, um eine genau Bestimmung zu erlauben. Die semistrukturierten Interviews behandeln nach der Literatur hier größere Problembereiche.
Interviews mit einem bestimmten Fokus (Strukturierte bzw. Semistrukturierte) müssen sehr gut vorbereitet sein und bedürfen mehr Pausenzeiten als unstrukturierte, da sich bei der Befragung sowohl der Experte wie die Fragenden sehr konzentrieren müssen und daher schnell ermüden. Die Ergebnisse vorbereiteter Interviews können dafür leichter in die Wissensbasis überführt werden, weil sie normalerweise weniger Redundanzen erhalten können als unstrukturierte.

Beim teachback Interview wechseln Experte und Wissensingenieur die Rollen. Hier erklärt der KE dem Experten was er verstanden hat und kann somit eigene Missverständnisse ausräumen.
Beim tutoriellen Interview hält der Experte einen Vortrag über bestimmte Problembereiche. Dabei ergibt sich aber die Schwierigkeit, das Experten zwar auf direkte Befragungen gut antworten können, eine eigenständige Strukturierung eines Vortrags für sie nicht immer leicht ist. Diese Interviewtechnik sollte zeitlich nicht übertrieben werden und sich eher auf ganz spezielle Fälle beschränken.
Bei der Fallbesprechung wird ein interessanter Fall des Problembereichs diskutiert. Man sucht hier vor allem zum Anfang der Befragungen besonders anschauliche Beispiele heraus, um den Experten zu motivieren. Es ist anzuraten dies innerhalb der Befragungen immer wieder locker einzustreuen, um die manchmal doch sehr trockene Aufbauphase, in der auch viele Trivialfehler untersucht werden müssen, immer wieder aufzulockern.
Allerdings werden Fallbesprechungen, wenn sie am Anfang der Erhebungen stattfinden, wahrscheinlich vom KE noch nicht richtig eingeordnet werden können, weil ihm der hierfür notwendige Überblick noch fehlt.
Jedes „ Interview ist aber mehr als nur das Stellen vorbereiterer Fragen“/Knill 03/. Der interessierte Leser sei hier auf die URL /Knill 03/ hingewiesen, in der verschiedene Bausteine dargestellt werden, die hier zu beachten sind.
Beim „lauten“ Denken (Protokoll-Analyse) schildert der Experte dem Wissensingenieur seine verschiedenen Problemlösungsschritt und die Überlegungen, die er dabei hatte. Man unterscheidet hier das gleichzeitige (concurrent) und das retrospektive Protokoll. Beim gleichzeitigen Protokoll schildert der Experte sein Vorgehen während des Prozesses, beim retrospektiven nach Abschluss des Vorgangs.
Das Problem beim gleichzeitigen Schildern und Ausführen einer Problemlösung ist es, dass der Experte möglicherweise durch den Erklärungsvorgang abgelenkt wird, weil normalerweise viele Expertenaktivitäten eher unbewusst sind. Möglicherweise kann der Experte solche Vorgänge auch nur schwer beschreiben.
Nach Abschluss des Vorgangs das Getane zu erklären birgt hingegen die Gefahr, dass bestimmte Details einfach nicht genannt werden, weil sie vom Experten bei der Erklärung als unwichtig erachtet werden oder weil sie dem Experten selbst nicht bewusst sind.
Ein weiteres Problem ergibt sich durch eine Interaktion des KE. Nachfragen würden den Experten ablenken, so dass eine konkrete Einordnung in die Wissensbasis erst im Anschluss erfolgen kann.

Bei der Feldbeobachtung beobachtet der Experte den Experten bei der Arbeit, ohne ihn zu befragen. Dies kann mit Wissen des Experten direkt durch die Anwesenheit des Experten im Feld erfolgen, kann aber auch über die Analyse von Videoaufnahmen vorgenommen werden. Diese Aufnahmen könnten sogar vom Experten direkt gemacht werden, um bestimmte typische Problemlösungsvorgänge zu dokumentieren, und in der Wissensbasis weiterverwendet werden.
Die Gruppendiskussion kann in der Erhebung meiner Meinung nach ganz zu Anfang, während der Auswahl eines geeigneten Experten, oder zur Überprüfung der entstandenen Wissensbasis durchgeführt werden. Die Arbeit mit mehreren Experten parallel, die an einer Wissensbasis arbeiten, halte ich hingegen für eher problematisch, zu dem wird es nur wenige Kunden geben die gleichzeitig mehrere Experten gleicher Güte besitzen und diese dann auch noch für die Arbeiten freistellen.
Brainstorming kann an verschiedenen Stellen der Erhebung angewandt werden, um neue Ideen in den Verarbeitungsprozess einzubringen, wobei sowohl auf die Wissensbasis wie die Mensch/Maschine Schnittstelle fokussiert werden sollte. Dabei können die Nutzer einbezogen werden, um so auch allmählich eine Akzeptanz des Systems bei diesen zu erreichen. Können deren Anregungen auch aufgenommen werden, wird sich die Einführung des XPS wesentlich erleichtern.
Im Review wird die Arbeit des Expertensystems mit einem oder mehreren Experten überprüft. Dazu werden bestimmte Problemfälle simuliert. Ein weiterführender Test durch nicht unmittelbar an der Erstellung beteiligten Experten oder Nutzer ist hier sinnvoll, weil der Experte zu sehr mit einer bestimmten Erwartungshaltung in die Problemlösung hineingeht und daher mögliche Fehleingaben vielleicht gar nicht überprüft.
Die Simulation dient auch dazu dem Experten den Arbeitsfortschritt zu zeigen und dazu mit dem Experten die verschiedenen Problemlösungsschritte durchzugehen, um Missverständnisse zwischen Experten und Wissensingenieur auszuräumen. Sie kann ab einem gewissen Reifegrad der Wissensbassi auch fester Bestandteil einer Sitzung werden. Die Dokumente können auch verwendet werden, um die entsprechende Fachsprache (siehe Ontologie) zu entwickeln, die in der Problemlösung verwendet wird.
Die Introspektion (aus der Psychologie entlehnter Begriff, lat. *die* Selbst-Beobachtung, die nach innen, das heißt auf das eigene Bewusstseinsgeschehen, gerichtete Beobachtung. /Brockhaus 10/), bezeichnet hier eine Selbstaufschreibung des Experten, die z. B. bei der Bearbeitung eines

bestimmten Fehlerfalls in der Diagnostik verwendet werde kann. Der KE muss bei diesem Vorgang nicht zugegen sein.
Die schriftliche Befragung von Experten durch den Wissensingenieur hat den Vorteil, dass sie in einer stillen Stunde vom Experten bearbeitet werden kann und ihn nicht allzu sehr behindert. Die entstandenen Unterlagen können dann wieder im Rahmen eines Interviews mit dem KE durchgearbeitet werden.
Innerhalb der Delphi-Methode werden Gruppenmeinungen hinterfragt, wobei die Experten hier anonym bleiben können. Dabei werden die Experten in mehreren Befragungsrunden wiederholt mit den verdichteten Ergebnissen der Befragung konfrontiert, um einen Konsens zu erreichen (mehrstufiges Ermittlungsverfahren). Damit sollen widersprüchliche Expertenmeinungen abgeklärt werden /Wagner 93/. Führt die Methode nicht zum Erfolg kann sie immerhin die Grundlage für weiterführende Interviews darstellen.
Umfasst der Problemlösungsvorgang Interaktionen des Experten mit anderen Personen (etwa Kunden oder Zulieferer), so können die hier geführten Dialoge ausgewertet werden (etwa durch Auswertung von aufgenommenen Gesprächen oder durch Analyse von Ergebnisprotokollen).
Mit Soziometrie wird die Beziehung der Mitglieder einer Gruppe untersucht /Wagner 93/. Die Problemlösung, die in einem Expertensystem abgebildet ist, bezieht gewöhnlich eine Gruppe von Menschen mit ein, die daran arbeiten. So sind zum Beispiel in der technischen Diagnose einer Transferstraße Elektriker, Schlosser, Steuerungsprogrammierer und Anlagenführer involviert.

c) Durchführung
Die Gesprächsführung des Experten muss sich an der Struktur des Problembereichs orientieren, also bei der technischen Diagnose am Aufbau der zu diagnostizierenden Anlage (Hierarchisierung des Problembereichs). Dabei sollte diese Struktur systematisch und ohne große Sprünge erarbeitet werden, um den Überblick nicht zu verlieren. Damit können die Lücken und Inkonsistenzen innerhalb der Wissensbasis wenn nicht vermieden, so doch reduziert werden.
Um eine reale Wissensbasis aufzubauen, ist es notwendig sich zunächst mit den Experten ein Konzept zu erarbeiten, in welchem Teile des Problembereichs in ihrer Bedeutung identifiziert werden sollten. Dadurch ist es möglich die Wissensbasis auch unterschiedlich fein auszuarbeiten. Es ist zum Beispiel in der technischen Diagnostik überhaupt nicht sinnvoll alle Teile einer Anlage mit der gleichen Genauigkeit zu betrachten. Teile, die sehr fehleranfällig sind, sollten hier mehr Beachtung geschenkt werden, als solche die erfahrungsgemäß wenige Fehler verursachen. Daneben sollte auch die

Austauschtiefe beachtet werden. Teile, die also im Fehlerfall einfach ausgetauscht werden, sollten auch nicht weiter untersucht werden.
Ein solches Vorgehen erhöht meiner Erfahrung nach auch die Akzeptanz bei den Experten, denen es ansonsten scher begreiflich zu machen ist, warum man sich mit Teilen beschäftigt, die noch nie defekt waren. Diese „sinnvollen Löcher" einer Wissensbasis können dann im Verlaufe der Akquisition immer noch ausgefüllt werden, wenn sich unvorhergesehene Hinweise ergeben sollten.
Der Knowledge Engineer muss sich am Vorgehen des Experten orientieren und daraus die Fakten ableiten, ohne diese selbst zu sehr zu interpretieren. Er muss sich beständig davon überzeugen, dass er die Fakten richtig aufgenommen hat (teaching interviews).
Verwendet der Experte bisher unbekannte Begriffe, so sollten diese direkt in ein Begriffslexikon aufgenommen und definiert werden. Damit wird es dem KE ermöglicht die Argumentation des Experten zu verstehen, was nur erfolgen kann wenn beide die gleiche Begriffswelt benutzen.
Innerhalb der Wissenserhebung kommt es auch vor, dass der Experte Zusammenhänge erkennt, die ihm bis dahin nicht bewusst waren. Dies stellt einen nützlichen Nebeneffekt der Wissenserhebung dar, der die Arbeit des Experten auch ohne XPS verbessern kann.
Die Auswahl der in der Erhebung verwendeten Beispiele spielt nach /Bachelier 04/ eine beachtliche Rolle. Er unterscheidet hier zum Beispeil

- typische Fälle, die häufig vorkommen und hohe Relevanz besitzen,
- Fälle mit limitierten Informationen, in denen beobachtet werden soll wie der Experte Strategien und Heuristiken einsetzt, um dem Informationsdefizit zu begegnen und
- Fälle mit Zeitbeschränkungen für die Lösung, wobei das vom Experten als wichtig erachtete oder besonders zielführende Vorgehen ermittelt werden soll.

Auch die Einbringung von Varianten zu dem durch den Experten aufgeführten Wissen durch den KE kann sinnvoll sein, wenn diese praxisrelevant sind. Dazu muss der Wissensingenieur aber erst in den Problembereich gut eingearbeitet sein.
Um sehr konzentriert Wissen aufnehmen zu können, hat das Einzelgespräch gegenüber dem Gruppengespräch den Vorteil, dass man sich dabei vollständig auf die Person konzentrieren kann, wohingegen ein Gespräch mit mehreren

Experten auch schon einmal in Unstimmigkeiten zwischen diesen münden kann.
Auf die in der Literatur / Universität Würzburg 03/ /Wagner 93/ ebenfalls vorkommenden Erhebungstechniken, wie etwa das Konstruktionsgitterverfahren oder Hierarchisches Clustering möchte ich hier nicht näher eingehen, weil sie meiner Meinung nach nur für sehr kleine Wissensbasen oder für Teilbereiche eventuell sinnvoll sein könnten. Dabei werden - sehr vereinfacht dargestellt – Ähnlichkeiten und Unterschiede zwischen Wissensbestandteilen systematisch aufgebaut. In den Beispielen hierzu wird aber immer nur eine sehr kleine Anzahl von Objekten betrachtet. In einer realen Wissensbasis befinden sich aber tausende von z.B. Diagnosen. Ähnlichkeiten und Unterschiede zwischen diesen zu erarbeiten mag akademisch reizvoll sein, sprengt normalerweise aber den vorgesehenen Arbeitshorizont bei weitem. Der interessierte Leser sei hier auf die oben angegebene Literatur verwiesen.
Um den Experten während der sich sehr lange hinziehenden Erhebungsphase motiviert halten zu können, empfiehlt es sich eine Mischung der beschriebenen Methoden zu verwenden /Computerwoche 87/.

6.2 Wissensmodellierung

Das vom Wissensingenieur innerhalb der Wissenserhebung aufgenommen Wissen muss in eine Form überführt werden, dass es durch das XPS verwendet werden kann. Dieser Vorgang wird in der Literatur auch als Formulierung oder Modellierungsphase bezeichnet /Wagner 93/.Innerhalb dieses Umsetzungsvorgangs muss der KE:

- sich selbst über den Problemlösungsvorgang klar werden und sich darüber ein mentales Modell erschaffen.
- eine Begriffswelt aufbauen, die die vom Experten verwendeten Begriffe und deren Zusammenhang festlegt,
- das erhobene Wissen in die vorher gewählte Repräsentationsform überführen und
- eine Strategie für die Problemlösung festlegen.

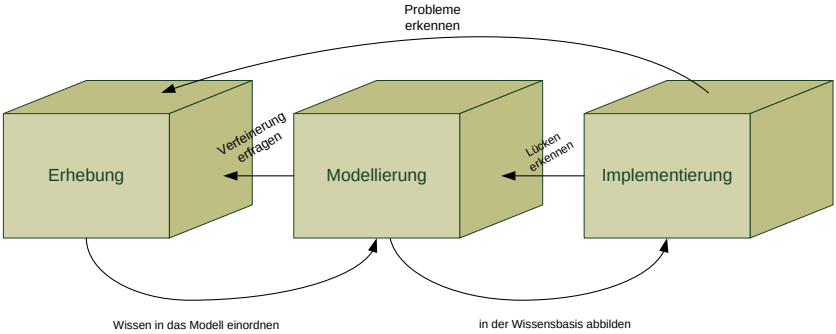


Abbildung 6.4: *Erhebung – Modellierung - Implementierung*

6.2.1 mentale Modellbildung

In der Literatur (z.B. /Wiegand 04/) wird der transferorientierte und der modellierungsorientierte Ansatz unterschieden.
Als transferorientierter Ansatz wird dabei verstanden, dass der Experte systematisch ausgefragt wird und das Wissen dann direkt in die Wissensbasis übertragen wird. Der Ansatz geht davon aus, dass das Wissen erhoben und ohne Veränderung übernommen werden kann.
Beim modellorientierten Ansatz wird zunächst ein Expertisemodell entwickelt, dass den Problemlösungsvorgang und das verwendet Wissen in einem Modell zusammenführt und die Wissensbasis entsprechend diesem Modell aufbaut.
Kein Knowledge Engineer, der auch nur geringe Erfahrungen hat, würde den so beschriebenen reinen transferorientierten Ansatz verwenden und ich bezweifele auch ob er jemals verwendet wurde. Jeder Mensch, der Wissen abbildet, ordnet das Wissen entsprechend einem Schema an, dass sich daraus ergibt wie er selbst den Problemlösungsvorgang verstanden hat. Damit aber wird ein Modell gebildet.
Noch unverständlicher empfinde ich es, dass der transferorientierte Ansatz und das Rapide Prototyping (siehe Kapitel 6.4) in der Literatur vermischt werden. Für mich ist das Rapide Prototyping eine Implementierungs- und Evaluierungsansatz und keine Modellierung. Ich könnte auch ein durch einen modellorientierten Ansatz erstellte Wissensbasis mit Rapide Prototyping

implementieren. Die Modellierung findet einfach auf einer anderen Ebene statt.

Jeder ernsthafte Wissensingenieur beginnt seine Tätigkeit mit der Bildung eines mentalen Modells, das es ihm ermöglicht den Problembereich für sich selbst verständlich zu machen. Dieses mentale Modell wird er im Laufe des Knowledge Engineerings ständig weiter verfeinern und in der Wissensbasis auch abbilden.

Als mentales Modell wird die Repräsentation eines Weltausschnitts im Bewusstsein eines Lebewesens verstanden. Da das Lebewesen von seinen Perzeptoren abhängig ist, wird dieses innere Modell der Außenwelt immer eingeschränkt und vereinfacht sein (vergleiche hierzu auch Kapitel 3 – Wissen). Gute mentale Modelle ermöglichen es dem Menschen die Komplexität der Welt zu reduzieren und sie damit beherrschbar zu machen. /wiki 10 c//Wiegand 04/

Als Beispiele für Mentale Modelle können das Herz als Pumpe oder die Assoziation eines stromführenden Kabels mit einem Wasserschlauch dienen. Jedes Modell hat aber auch seine Grenzen, so wird der Wasserschlauch zwar dem Widerstand eines Stromkabels veranschaulichen können, wird aber ein gänzlich anderes Verhalten beim Durchschneiden zeigen als der stromdurchflossene Leiter. /Blumstengel 98/

Da kein Lebewesen die gleiche Wissensstruktur aufgebaut hat, sind alle mentalen Modelle grundsätzlich individuell, wir sind es aber gewohnt sie auf den Erfahrungsschatz unserer Gemeinschaft umzusetzen, so dass sie von vielen Mitgliedern unserer Gesellschaft verstanden werden können. Einem Menschen, der keine Pumpe kennt, wird auch der Vergleich Herz-Pumpe keine zusätzlichen Informationen bringen können.

Mentale Modelle ermöglichen es uns durch die Vereinfachung Sachverhalte leichter zu verstehen und erleichtern uns damit die Planung unserer Handlungen /Hasebrook 95/. Innerhalb des Knowldege Engineerings wird das Modell dazu verwendet den Problemlösungsvorgang zu strukturieren. Dabei wird das Modell im Laufe des Knowldege Engineerings immer weiter ausgebaut. Dieser Vorgang ist damit dem flexiblen Wissensstrukturierungsvorgang des Menschen ähnlich, der auch Rücknahmen und Umstrukturierung zulässt. /Eisfeld 01/

Der Vorgang des Ausfüllens des Modells mit dem durch den Wissensingenieur interpretierten und den Anforderungen der spezifischen Repräsentationsform entsprechenden Wissen wird auch als Wissenssoperationalisierung bezeichnet/wiki 10d/. Innerhalb dieses Vorgangs werden mentale Operationen, wie Klassifikation oder Sortierungen angewendet, um das

aufgenommene Wissen richtig einordnen zu können. /lexikon stangel 10/ / Wachsmuth 94//Universität Würzburg 03//Wiki 10/

Mit der schrittweisen Abbildung des Wissens in einem Modell sollen folgende Ziele verfolgt werden:

- Schnellere Einordnung neuer Wissenselemente, wenn sie in das vorhandene Modell passen,
- leichtere Auffindung fehlender Wissensbestandteile,
- leichtere Entdeckung von Inkonsistenzen und Unvollständigkeiten innerhalb der Wissensbasis./Bachelier 04/

In der Literatur werden zwei verschiedene modellbasierte Ansätze erwähnt:

- Generic Tasks, in dem davon ausgegangen wird , dass alle Expertenaufgaben durch eine kleine Anzahl von Problemlösungsverfahren beschreibbar sind und
- Integrierte Modelle, bei welchem man von hierarchisch modellierbaren Ebenen ausgeht, wobei nur die unterste domänspezifisch ist. /Bachelier 04/

Den ersten Ansatz halte ich nur für sehr kleine Problemlösungsbereiche für sinnvoll, außerdem stellt sich hierbei das Problem welchen Problemlösungsansatz man auswählt, ob alle äquivalent sind und ob man überhaupt alle Verfahren entdeckt hat.
Auf die integrierten Modelle, etwa KADS wird in 6.3 eingegangen. Sie stellen mächtige Werkzeuge dar, die vielleicht aber schon für viele Anwendung überdimensioniert zu sein scheinen.
Bereits in der Wissenserhebung muss ein gemeinsames mentales Modell geschaffen werden, weil der einzelne Experte normalerweise eine Vorstellung über das Problemfeld hat, die den Knowledge Engineers aber auch anderen beteiligten Experten nicht unmittelbar zugänglich ist. Innerhalb des Projekts WAVES untersucht z.B. die Forschungsgruppe Technik und Gesellschaft im Forschungsverbund *Lebensraum Stadt* verschiedene Aspekte Zusammenarbeit solcher heterogener Gruppen. /Waves 02/
Bei der Umsetzung vom mentalen Modell des Experten kommt es nach /Herczeg 05//Herczeg 06/ laufend zu Entscheidungen, die im Nachhinein nur schwer nachvollziehbar sind und die beim Benutzer, Systementwickler und Experte verschiedene Vorstellungen vom eigentlich abzubildenden System erzeugen. Herczeg untersucht dabei die verschiedenen Mensch-Maschine-

Systeme und bildet Modell verschiedener Ordnung ab, die er detailliert untersucht. Der Vorgang der Systemabbildung und Modellierung unter dem hier angegebenen Blickwinkel wird von Herczeg als Cognitive Systems Engineering bezeichnet.

6.2.2 Ontologie

Die Realisierung eines Modells geht mit der Schaffung einer Terministruktur (Ontologie) einher, die für den Experten verständlich ist und die eine Verständigung zwischen KE und Experten im Problembereich garantiert.

Der Begriff Ontologie entstammt eigentlich der Philosophie, wo er seit dem 18. Jahrhundert die allgemeine Metaphysik, die Lehre vom Sein, bezeichnet. In dem hier betrachteten Sinne stellen Ontologien Fachterminologien dar, die zur Verständigung von Personen notwendig sind, die an einem Arbeitsprozess beteiligt sind. Nur durch die Verwendung von einheitlichen Begriffen ist der Austausch von Daten, deren Wiederverwendung und Verifikation überhaupt erst möglich. Ontologien können dabei sowohl als Objektsprache wie als Metasprache auftreten. In der Logik ist sowohl die, die logischen Operationen beschreibende Sprache, wie die Metasprache, welche zur Beschreibung logischer Schlussfolgerungen verwendet wird, als Ontologie bzw. Metaontologie aufzufassen. /Schlieder 01/
Nach /Lutz 02/ stellt eine Ontologie eine Formalisierung eines bestimmten Weltausschnitts dar. Sie besteht dabei aus einer Menge von Begriffen (Vokabular) und einer Beschreibung ihrer Bedeutung. Diese Beschreibungen sollen Definitionen enthalten und die Beziehung zwischen den verschiedenen Bestandteilen klarmachen. Sinn der Ontologie ist es die mögliche Interpretation verschiedener Begriffen einzugrenzen und die Struktur des betrachteten Weltausschnitts zu erfassen. Damit stellt sie ein wichtiges Verständigungsmittel zwischen dem Experten und dem KE dar, ermöglicht also eigentlich erst eine gleiche Weltsicht in bezug auf den Problembereich. /Wiegand 04/

Ontologien lassen sich in verschiedene Abstraktionsebenen unterteilen, die aber ineinander übergehen können. So unterscheidet man /Lutz 02/:

- Top Level oder Upper Ontologien

1. Hier werden allgemeine Elemente wie „Objekt“ oder „Aktivität“, die in vielen verschiedenen Bereichen gleich verwendet werden, beschrieben.

- Domain Ontologien
  2. Diese beziehen sich auf einen beschränkten Weltausschnitt etwa Maschinenbau oder Produktionstechnik
- Task-, Methoden- oder Problemlösungs-Ontologien
  3. Beschreiben allgemeines Problemlösungswissen, das in unterschiedlichen Bereichen verwendet werden kann
- Anwendungs-Ontologien
  4. Schränkt die Domain- und Task-Ontologie auf einen bestimmten Anwendungsbereich ein.

Einmal aufgestellte Ontologien lassen sich auch in anderen verwandten Problembereichen verwenden. Um diese Wiederverwendung leicht möglich zu machen, ist es notwendig den Kontext der Ontologie darzustellen und sie leicht zugänglich zu machen. Dazu haben sich sogenannten Ontologie-Bibliotheken etabliert (z.B. gibt Lutz hier mit Verweis auf weitere Literaturstellen den URL http://www-ksl-svc.stanford.edu:5915 für den KSL-Ontology Server an).
Damit stellen Ontologie eine Möglichkeit dar bereits analysiertes Wissen wiederverwendbar zu machen und besitzt darüber hinaus gegenüber herkömmlicher Datenbankanwendung auch im Bereich der Syntax und Semantik Vorteile. (siehe hierzu auch /Wiegand 04/)

Die seit Anfang der 90zieger Jahre in der KI eigesetzten Ontologien stellen eine begriffliche bzw. natürlichsprachliche Vorstrukturierung des Wissens dar. Die Bedeutung und Verwendung der natürlichsprachlichen Ausdrücke wird durch semantische Regeln festgelegt. /Wiegand 04/
Die wachsende Bedeutung der Ontologien, z.B. auch für das Semantische Web und im Bereich des Wissensmanagements machen heute auch Text-Mining-Verfahren /Kunze 03/, die aus großen Datenbeständen relevante Ontologie-Bestandteile identifizieren können, für das ontology engineering interessant. /Wolff 05/

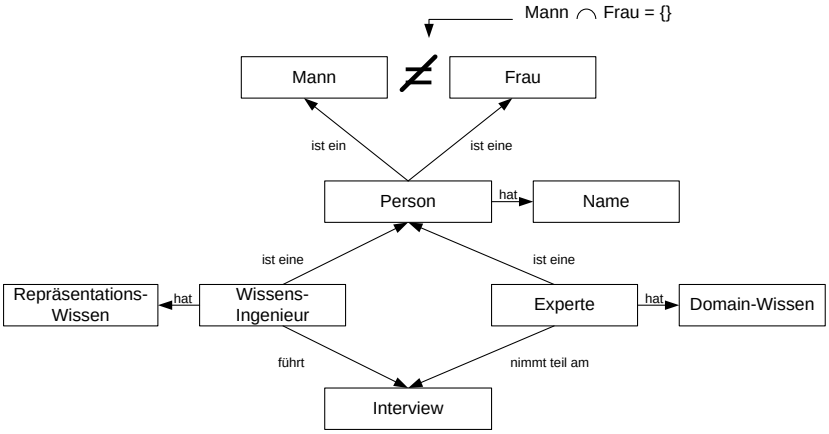


Abbildung 6.5: *Beispiel Ontologie*

6.2.3 Wissensrepräsentation

Die Wissensrepräsentation ist immer problemadäquat zu wählen, sie richtet sich also danach was man abbilden möchte. Eine einzige Repräsentationsform für alle möglichen Problemfelder gibt es nicht, diese richte sich vielmehr am Problem, dem Benutzerkreis und dem geplanten Erstellungs- und Wartungsaufwand aus /Wiegand 04/. Durch die in der Ontologie bereits stattfindende Vorstrukturierung des Wissens, ist es möglich zu erkennen welche der verschiedenen Wissensrepräsentationsformen man verwenden sollte. Dabei gibt es Formen, denen nur ein sehr eingeschränkter Einsatz zukommt, etwa Constraints und andere, welche praktisch überall eingesetzt werden können etwa Regeln und Objekte.
Gerade bei der Verwendung von XPS-Shells und Toolboxen sollte man auf die unterschiedliche Probleme bei der Wissensrepräsentation achten und Entwicklungsumgebungen bevorzugen, die ein möglichst weites Spektrum an Möglichkeiten bieten.
Hat man sich für eine problemadäquate Repräsentationsform entschieden, ist noch zu klären wie das modellierte und vorstrukturierte Wissen in eine durch

das System verwendbare Wissensbasis übertragen werden kann. Die Repräsentation sagt nur, dass Regeln, Objekte oder Tabellen verwendet werden sollen und nicht wie der Übersetzungsvorgang stattfinden soll.
In meiner Praxis bewährt hat sich hier die Erstellung einer Wissensrepräsentationssprache (WRS), die auch für den Experten lesbar und mit einem einfachen Textsystem manipulierbar ist. Diese WRS kann sich an der bereits erstellten Ontologie orientieren und ist idealerweise auch direkt innerhalb des Vorgangs der Ontologie-Erstellung mit dem Experten zu kreieren.
Werden Entwicklungswerkzeuge zur Erstellung der Wissensbasis verwendet, so kann es allein schon dadurch zu Problemen im Verständnis des Experten kommen, dass diese Werkzeug eine bestimmte Syntax benutzen, die möglicherweise nicht problemadäquat sein kann. Da ein Werkzeug für viele verschiedene Anwendungsfälle verwendbar sein soll, kann es nicht genau auf ein spezielles Problem angepasst sein, wie dies bei einer Einzelanwendung der Fall ist.
Die mit Hilfe der WRS erstellte externe Repräsentation der Wissensbasis liegt in Form einer lesbaren Textdatei vor und kann über ein als Parser bezeichnetes Programm in eine interne Form verwandelt werden, auf die dann der eigentliche Problemlösungsmechanismus zu greift.
Die externe Präsentation eignet sich durch ihre Textform auch als ein von einem Manager (z.B. dem Vorgesetzten des Experten) verstehbares Projektfortschrittsdokument und könnte sogar mit einem entsprechenden Umsetzer von ganz verschiedenen XPS-Werkzeugen verwendet werden. Beachtet man den erheblichen Aufwand, den die Erstellung einer Wissensbasis verursacht sind solche Aspekte durchaus auch mit zu berücksichtigen.
In Kapitel 5 wird eine Wissensrepräsentationssprache für den Fall einer technischen Diagnose vorgestellt.

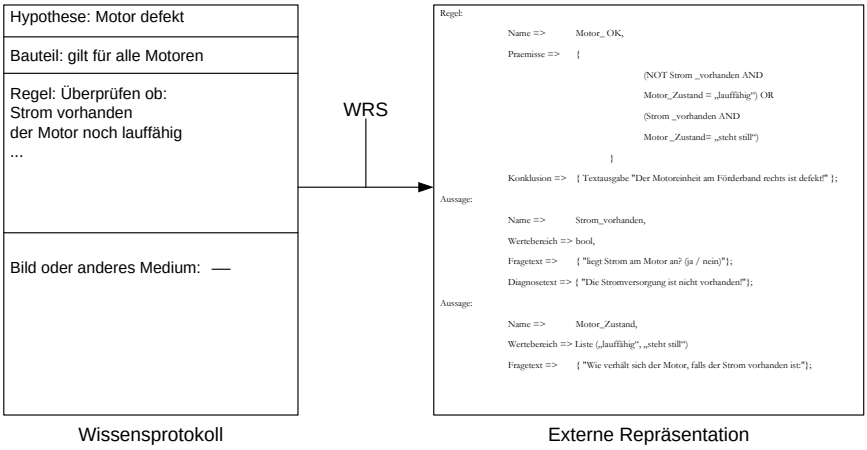

Hypothese: Motor defekt
Bauteil: gilt für alle Motoren
Regel: Überprüfen ob:
Strom vorhanden
der Motor noch lauffähig
...
Bild oder anderes Medium: —
WRS
Regel:
Name => Motor_ OK,
Praemisse => {
(NOT Strom _vorhanden AND
Motor_Zustand = „lauffähig") OR
(Strom _vorhanden AND
Motor _Zustand= „steht still")
}
Konklusion => { Textausgabe "Der Motoreinheit am Förderband rechts ist defekt!" };
Aussage:
Name => Strom_vorhanden,
Wertebereich => bool,
Fragetext => { "liegt Strom am Motor an? (ja / nein)"};
Diagnosetext => { "Die Stromversorgung ist nicht vorhanden!"};
Aussage:
Name => Motor_Zustand,
Wertebereich => Liste („lauffähig", „steht still")
Fragetext => { "Wie verhält sich der Motor, falls der Strom vorhanden ist:"};
Wissensprotokoll
Externe Repräsentation

Abbildung 6.6: *vom Wissensprotokoll zur Repräsentation*

6.2.4 Strategien für die Problemlösung

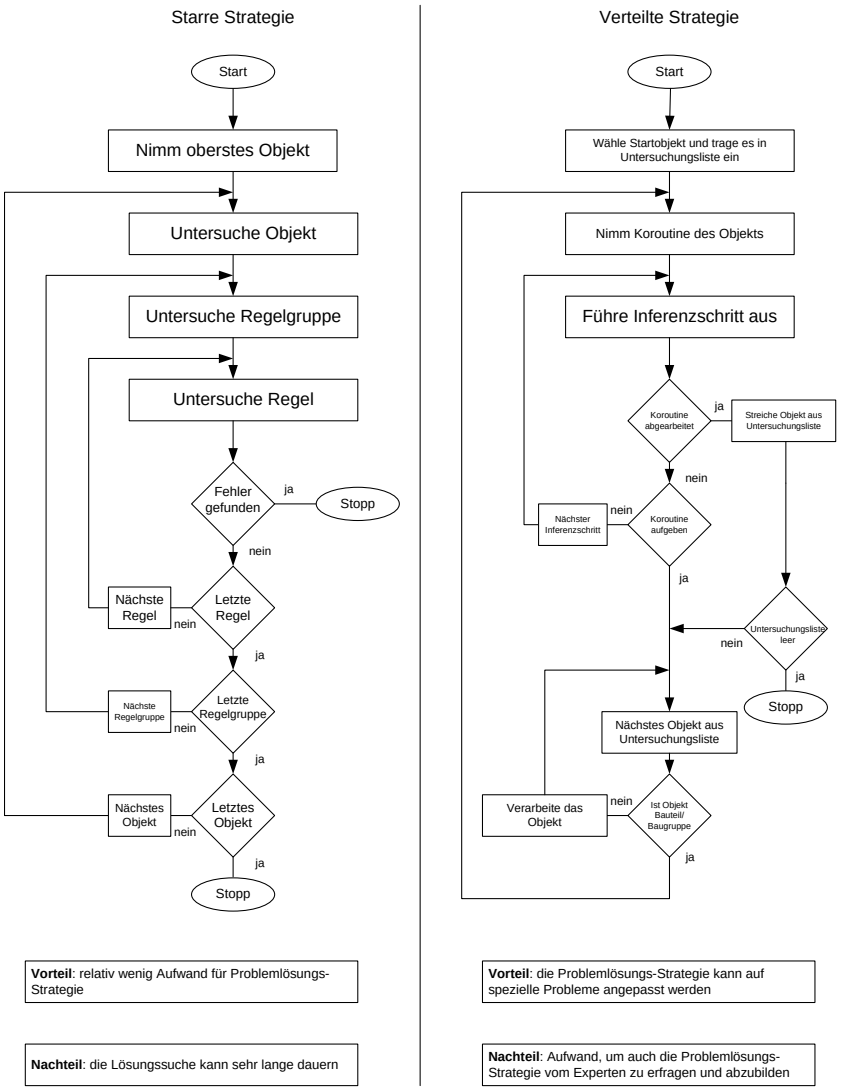
Schon während der Wissenserhebung wird der Experte seine Problemlösung in einer bestimmten Reihenfolge von Lösungsschritten vornehmen. Der KE muss diese Lösungsschritte in irgendeiner Form nachvollziehen, ist dabei aber abhängig von der Repräsentationsform die gewählt wurde.
Bei der Verwendung der herkömmlichen Repräsentationsformen war die Bestimmung der Strategie nur implizit, etwa über die Verarbeitungs-Reihenfolge der Wissensbestandteile z.B. der Regeln oder Objekte möglich. Auch die Umsortierung oder die Bildung von Untermengen von Wissenselementen war als mögliche Abbildung von speziellem Problemlösungsverhalten möglich.
Bei der Entwicklung der von mir mit realisierten XPS gingen die Entwickler allerdings einen anderen Weg. Um die Problemlösungsschritte eines Experten möglichst realistisch abbilden zu können wurden:

- neue Strukturierungsmittel eingeführt, so wurden Regeln zu Regelgruppen zusammengefasst und damit leichter verarbeitbar. Prinzipiell konnte dabei eine Regel sogar mehrere Regelgruppen

zugeordnet werden. Die Regelgruppen waren über Namen ansprechbar und wurden im Verlaufe der Wissenserhebung vom Experten einem bestimmten Objekt zugeordnet. Damit war es zum Beispiel möglich, eine bestimmte Fehlerhypothese mit einer bestimmten Regelgruppe zu untersuchen. Alle Regeln, die auf diese Hypothese verwiesen, konnten also der Regelgruppe zugeordnet werden. Auch eine ganze Baugruppe kann untersucht werden, in dem das übergeordnete Element Baugruppe eingeführt wurde, welches die einzelnen Bauteile zu einer Gruppe zusammenfasst.

- die Verarbeitung explizit durch die Verwendung von Problemlösungsschritten bestimmt wurde. Diese Problemlösungsschritte stellen eine Verteilung des Problemlösungsvorgangs auf verschiedene Objekte dar. Der Experte konnte über die Verwendung dieser Schritte den Problemlösungsvorgang einzelner Objekte bestimmen und damit seine ganz spezielle Problemlösungsstrategie umsetzen. Prinzipiell kann ein Problemlösungsschritt allen Wissenselementen zugeordnet werden und die Verarbeitung damit für ganz genau dieses Element bestimmen. Durch die Verwendung der Problemlösungsschritte in den Konklusionen von Regeln wurde eine situationsabhängige Problemlösung möglich. Die Problemlösungsschritte wurde dabei so flexibel gestaltet, dass es auch möglich war ganz bestimmte Wissensobjekte innerhalb einer Problemverarbeitung zu berücksichtigen.
- neue übergeordnete Verarbeitungseinheiten etwa die Koroutinen oder der Hypothesenfokus geschaffen, welche die Inferenzschritte enthielten und über die die Problemlösung zeitlich gesteuert wurde.

Die hier zusammengefassten Strukturen werden in einem Beispiel in Kapitel 5 dargestellt. Dieses stellt aber nur eine Möglichkeit dar, vielmehr sollte auch eine verteilte Problemlösung, weil sie Metawissen darstell, immer problemadäquat sein. Sie ermöglicht es aber sehr flexibel vorzugehen und die Verarbeitungsstrategie auch als Teil der Wissenserhebung zu betrachten. Kein Experte kann einsehen, dass alle Elemente entsprechend einer einheitlichen Verarbeitungsphilosophie verwendet werden sollen, wie das durch die Abarbeitung zum Beispiel eines einheitlichen Regelinterpreters notwendig war.



Vorteil: relativ wenig Aufwand für Problemlösungs-Strategie

Nachteil: die Lösungssuche kann sehr lange dauern

Vorteil: die Problemlösungs-Strategie kann auf spezielle Probleme angepasst werden

Nachteil: Aufwand, um auch die Problemlösungs-Strategie vom Experten zu erfragen und abzubilden

Abbildung 6.7: *Vergleich der Problemlösung zum starren Vorgehen*

6.3 Wissensintegration

Nach der Wissensmodellierung muss das Wissen in das System integriert werden. Dazu stehen mit dem Parser und der Akquisitionskomponente eher traditionelle Bestandteile der XPS-Technologie und mit den Modellierungswerkzeugen mächtige aber zum Teil überfrachtete Mittel zur Verfügung. Die Modellierungswerkzeuge führen den Ersteller eines Expertensystems durch die verschiedenen Modellierungsebenen bis hin zum fertigen System, erscheinen mir aber in hohem Maße rein akademisch. In der Praxis führt ein solcher Aufwand zu einer Vervielfachung der Kosten zur Erstellung eines Systems und ist deshalb nach meiner Meinung zur Erstellung einer Shell oder eines Toolkits aber nicht für jedes System anzuraten.

Bei der Verwendung von Werkzeugen, wie sie für die meisten Anwendungen allein schon aus Kostengründen anzuraten ist, wird ebenfalls eine Modellierung wie im Kapitel 6.3 Anzuraten sein, sie sollte sich aber vielleicht einem abgespeckten Ansatz der Modellierungswerkzeuge bedienen.

Die Wissenseingabe sollte nur von Experten für die entsprechende Wissensrepräsentation, also vom KE selbst vorgenommen werden. Den Experten an der Eingabe zu beteiligen kann die Vorteile, die man sich mit einer disziplinierten Modellierung erarbeitet hat, wieder zunichtemachen. Die Eingabe sollte auch erst erfolgen, wenn die Wissensaufbereitung einen gewissen Reifegrad erreicht hat. Nur dann erscheint die Überprüfung einer Wissensbasis mit dem System als sinnvoll, weil nur dann die Wissensbasis in einem mit dem Benutzer gut abgeglichenen Stand sein kann und man sich über die Ontologie und die Modellierung hinreichend klar ist. Nichts ist für die Akzeptanz eines Systems schädlicher als die Verwendung einer Wissensbasis mit vielen Inkorrektheiten, die daraus resultieren, dass der KE noch bestimmet Teile des Problemsfeldes nicht voll durchdrungen hat.

6.3.1 Verwendung eines Parsers

Der Parser wurde bereits im Kapitel 4.2.4 beschrieben. Er setzt die externe Repräsentation, wie sie durch die Wissenserhebung und –Modellierung entstanden ist, in eine interne Repräsentation um, die vom System dann auch wirklich verarbeitet werden kann.
Durch das sequentielle Einlesen der Wissensbestandteile ergibt sich ein Problem bei der Überprüfung der verwendeten Namen. So können eventuell Objekte verwendet werden, die erst später in der Datei definiert werden. Eine Aussage in einer Regelprämisse wird möglicherweise erst danach beschrieben. Erst nach deren Definition aber kann der Wertebereich und damit die Korrektheit innerhalb der Prämisse überprüft werden. Solche Auflösungen von Namenkonflikten kommen an mehreren Stellen der Wissensbasis, etwa zwischen Regeln und Regelgruppen und Regeln und Objekten vor. Die Probleme im Einzelne lassen sich nur nach der Definition der Wissensrepräsentationssprache klären. Diese legt die einzelnen Abhängigkeiten durch Definition fest.
Die angesprochenen Probleme lassen die Verarbeitung der Wissensbasis in zwei Phasen ratsam erscheinen. In der ersten Phase werden die einzelnen Bestandteile eingelesen ihre syntaktisch Korrektheit überprüft und die Namen in einzelnen Tabellen vermerkt. Innerhalb dieses Prozesses wird dann auch festgestellt werden können ob Namen doppelt vergeben wurden, was man aus Übersichtlichkeitsgründen vermeiden sollte. Durch die Wissensrepräsentationssprache ist dabei eine Klassifikation der einzelnen Bestanteile, also ob es sich bei dem Element um eine Objekt, eine Regel, eine Aussage etc. handelt schon festgelegt. In der zweiten Phase der Verarbeitung können dann entsprechend aufsteigender Komplexität die eigentlichen Wissensbestandteile aufgebaut werden. In der in Kapitel 5 vorgestellten Wissensrepräsentationssprache würde man somit entsprechend der Reihenfolge

Aussagen – Regeln – Regelgruppen – Bauteile –Baugruppen –Koroutinen

beim Aufbau der internen Repräsentation fortschreiten.

Nur vom Parser für fehlerfrei ermittelte Wissensbasen sollten weiterverarbeitet werden. Große Wissensbasen weisen immer Inkorrektheiten auf, die relativ schwer zu finden sind und einen großen Testaufwand bedingen. Daher sollte wenigstens die syntaktische Korrektheit garantiert werden.

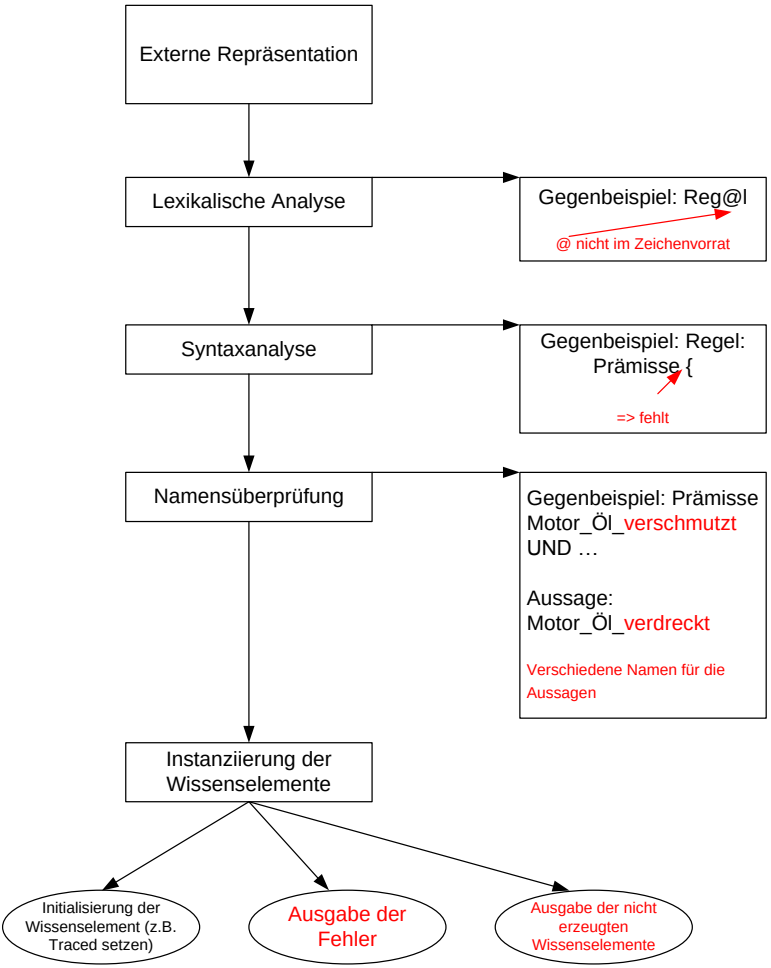


Abbildung 6.8: *Verarbeitungsschritte des Parsers*

6.3.2 Verwendung einer Akquisitionskomponente

Die im Abschnitt 4.3 und 4.5 in der Mensch-Maschine Schnittstelle dargestellten Wissenserwerbskomponente kann zur Erstellung von syntaktisch korrekten Wissensbasen verwenden werden.
Mit Hilfe eines syntaxgesteuerten Editors kann der Benutzer bei der Eingabe von Wissen bei der syntaktisch fehlerfreien Eingabe unterstützt werden. Dabei ergeben sich natürlich ebenfalls die oben angesprochenen Probleme der Reihenfolge bei der Eingabe von Wissensbestandteilen. Um die Syntax korrekt zu halten, werden dem Benutzer beim Verlassen eines Kontexts die noch nicht aufgelösten Referenzen angezeigt.
Durch die Verwendung einer Akquisitionskomponente kann es zu Wissensbasen kommen, die lückenhaft sind und die immer wieder vom Benutzer in ihrer Komplexität überschaut werden müssen. Tatsächlich birgt die Verwendung einer Akquisitionskomponente die Gefahr, dass der Knowledge Engineer, dem auch hier der Aufbau der Wissensbasis überlassen werden sollte, in Gefahr gerät den Überblick zu verlieren. Überlässt man dem System die Verfolgung der Korrektheit, geht diese möglicherweise nicht mit der gleichen Effizienz von statten, wie dies bei der Verwendung eines Parsers der Fall ist. Es besteht außerdem die Gefahr, dass neben dem Wissensingenieur auch andere Mitarbeiter an der Wissensbasis „herumschrauben“, weil dies durch den Editor leicht möglich wird.
Die Verwendung einer Akquisitionskomponente sollte ebenfalls erst dann vorgenommen werden, wenn die Modellierung einen gewissen Reifegrad erreicht hat. Die Inkorrektheiten in einer Wissensbasis sind meist nicht syntaktisch sondern semantisch und davor kann keine Wissenserwerbskomponente schützen.
Da eine Wissensbasis niemals am Stück aufgebaut werden kann, ist die Einbeziehung eines Parsers, der aus der internen Repräsentation wieder eine externe macht, notwendig, damit die Wissensbasis abgespeichert werden kann. Jede externe Repräsentation sollte dabei in einer lesbaren Form vorgenommen werden, um die Wissensbasis auch dokumentieren zu können. Die Abbildung in einer Datenbank halte ich persönlich als nicht sinnvoll, weil damit vielleicht eine effektive Abspeicherung erfolgen kann, die Lesbarkeit aber Schaden nehmen dürfte.

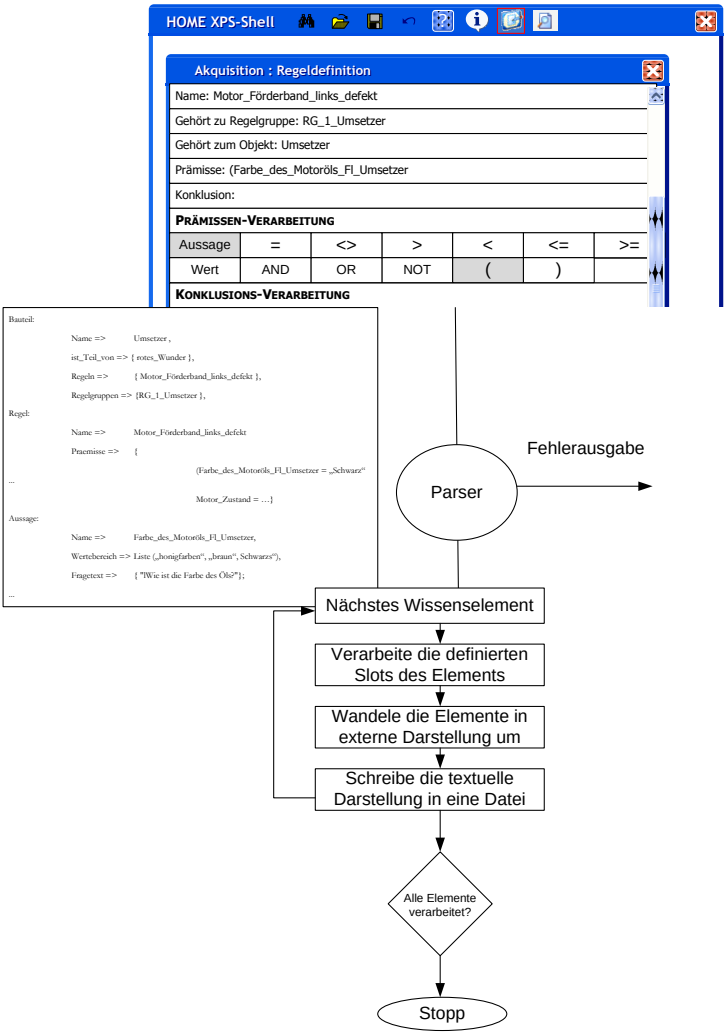


Abbildung 6.9: *Akquisitionskomponente und Parser*

6.3.3 Modellierungswerkzeuge

Die hier kurz andiskutierten Werkzeuge stellen mächtige Modellierungswerkzeuge dar, die von der Wissenserhebung bis zur Realisierung des Systems verwendet werden können /Wiegand 04/.

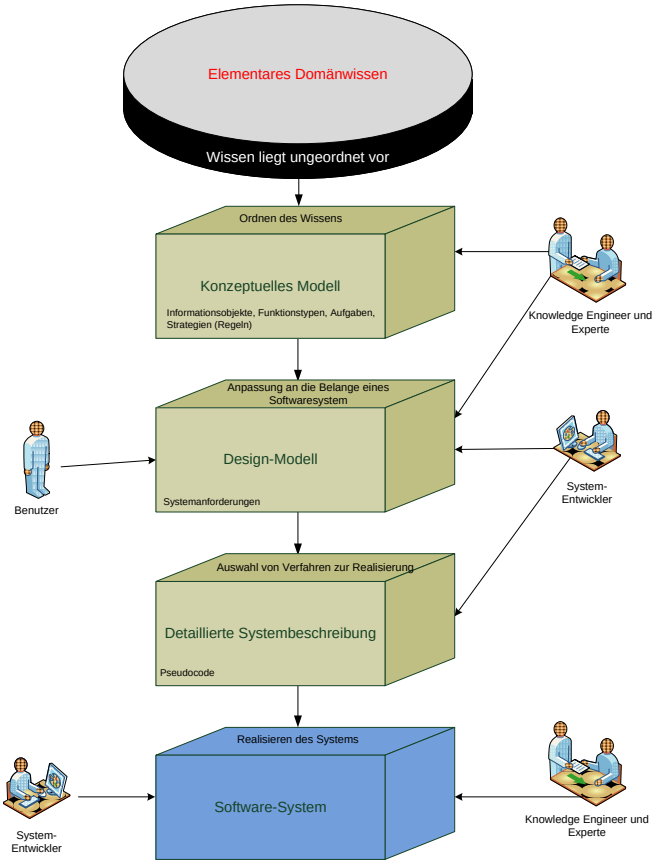


Abbildung 6.10: *KADS-Modellierung nach /Aue 90/*

Das auf den Arbeiten von Wielinga und Breuker (Universität Amsterdam) aus den 90ziger Jahren basierende System KADS(Knowledge Acquisition and Documentation Structuring), welches ab 1994 zu CommonKADS führte, stellt heute wohl das verbreiteste Modellierungs-Werkzeug dar /Wiegand 04/.
In der Literatur erfolgt die Beschreibung des Systems nicht ganz einheitlich, auch wenn das generelle Vorgehen überall weitgehend identisch beschrieben wird.
Nach /Bachelier 04/ unterscheidet KADS die folgenden Analyseebenen:

- Die Anwendungsebene,
  - in der Begriffe der Domäne, Relationen (is-a, has, causes) und Strukturen als Verknüpfungen zwischen diesen definiert werden (siehe hierzu auch Ontologien);
- Die Inferenzebene
  - Diese dient dazu Lösungsschritte (Knowledge-Source), Funktionen und den Datenfluss zu modellieren. Dabei werden Objekte der Anwendungsebene zu Meta-Klassen weiterentwickelt, die den Problemlösungsprozess steuern;
- Die Aufgabenebene,
  - In welcher Ziele und Teilaufgaben und die damit verbundene Problemlösungsstrategie festgelegt wird;
- Die Strategieebene,
  - Hier wird die strategische Metaebene der Problemlösung, wie Pläne und Metaregeln, festgelegt.

/Wiegand 04/ unterscheidet hier nur die Aufgaben- und Kontrollebene (Zusammenfassung der Aufgaben- und Strategieebene), Inferenzebene und Domänebene (Anwendungsebene). Inhaltlich unterscheiden sich die Ebenen aber nicht.
Durch die Einbeziehung einer auf UML basierenden Notation wird der KADS-Ansatz zum CommonKADS /Wiegand 04/. Damit wird eine Anbindung an das Wissens-Management möglich (siehe hierzu auch Wissens-Management-Systeme Kapitel 3.3)
/Aue 90/ geht bei der KADS-Modellierung von dem elementarem Domänwissen aus und entwickelt daraus ein Konzeptuelles Modell. In diesem Modell werden aus dem Domänwissen Klassen definiert und Informationsobjekte (Metaklassen) und Funktionstypen (Knowldege Sources) erzeugt. Daneben werden hier auch die Aufgaben (Task) und Strategien erzeugt. Um dieses konzeptuelle Modell einem Softwaresystem. Also dem eigentlichen XPS anzupassen, wird eine Design-Modell und aus diesem eine

detaillierte Systembeschreibung erzeugt, aus der dann der Programmcode des Systems entsteht.
Das konzeptuelle Modell wird hier dann wieder in den Domain Layer, den Inferenz-Layer, den Task Layer und den Strategy-Layer unterteilt. Als Hilfsmittel wird dabei eine erweiterte ERM (Entity-Relation-Model)-Darstellung verwendet (CommanKads Ansatz bei Wiegand).

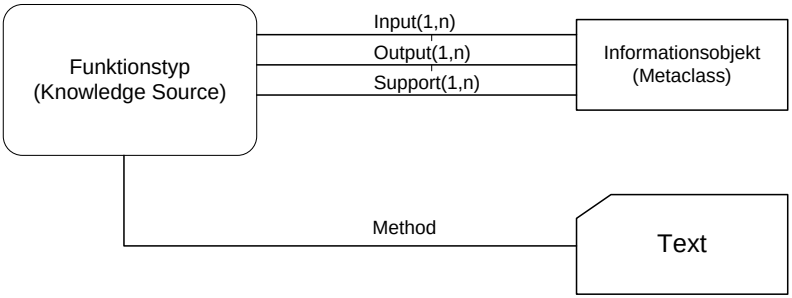


Abbildung 6.11: *Beispiel einer Darstellung mit ERM /Aue 90/*

Eine Forschungsgruppe um Studer entwickelte an der Universität Karlsruhe das System MIKE(Model-based and Incremental Knowledge Engineering). Die aus der Wissenserhebung resultierenden Wissensprotokolle, als eine natürlichsprachliche Zusammenfassung der Interviews und Erhebungen, stellen dabei den Anfangspunkt der Formalisierung dar. Durch ihre Interpretation werden die relevanten Daten, Strukturen und Relationen sowie die einzelnen Problemlösungsschritte abgeleitet, die dann als Grundlage für das sogenannte KARL-Modell dienen. Die verschiedenen Wissensbestandteile werden dabei durch die formale Sprache KARL (Knowledge Acquisition and Representation Language) dargestellt /Angele 91/.

Die beschriebenen Modelle können prinzipiell auch zur Erzeugung eines Unternehmensmodells verwendet werden. /Aue 90/ verwendet zum Beispiel einen Unternehmensmodellierungsansatz (URMEL), den er aus KADS ableitet. Dabei können für den Entscheidungsträger interessante Sichten modelliert werden, die sich gegenseitig ergänzen können. Aue unterscheidet hier zum Beispiel eine:

- Datensicht
  Im Unternehmensprozess anfallende Daten im Vordergrund
- Tätigkeitssicht
  Überblick der verschiedenen Tätigkeiten im Unternehmen

- Organisationssicht
  Organisationseinheiten und ihre Struktur und
- Entscheidungsschicht
  Zu treffende Entscheidung innerhalb des Unternehmensalltags.

6.4 Wissensevaluierung

Nach dem Aufbau der Wissensbasis muss diese in den Arbeitsablauf des Kunden integriert werden. Dieser Integration wird noch einmal eine umfassend Testphase vorausgehen, in dem der Experte und ausgewählte Nutzer die Problemlösung mit dem System überprüfen können. /Computerwoche 87/
Auch aufgrund der Größe einer realen Wissensbasis ist es aber nicht möglich mit dem Testen der Inhalte zu warten bis diese vollständig aufgebaut ist. Vielmehr werden immer wieder Teile des abgebildeten Wissens durch den Experten überprüft und innerhalb des Knowledge Engineering behandelt.
In der Literatur wird der Begriff des Rapid Prototyping sehr ambivalent behandelt. Zum Teil wird er als Gegensatz zum modellbasierten Ansatz erwähnt, zum Teil wird er als den transferorientierten Ansatz unterstützend betrachtet. Dann aber wieder wird in der Literatur auch beim modellbasierten Ansatz Prototyping erwähnt (z.B. / Lawson 97/).
Das Rapide Prototyping stellt nach meiner Meinung aber auch eine Möglichkeit der Evaluierung und der Integration einer Wissensbasis dar. Mir scheint die Überprüfung und schrittweise Integration vor allem bei großen Wissensbasen als sinnvoller als die Konfrontation der Nutzer mit fertigen Systemen. Der Gegenpart zum modellbasierten Ansatz stellt der transferorientierte Ansatz und nicht das Rapide Protoyping dar /Wiegand 04/, weshalb ich an dieser Stelle für den schrittweisen Aufbau des Systems und eine Integration werben möchte, nachdem eine Wissensmodellierung wie im Kapitel 6.3 beschrieben, vorgenommen wurde.
Die Evaluierung sollte von, nicht am Knowledge Engineering beteiligten Nutzern, vorgenommen werden. Sollten hier Schwächen in der Bedienung des Systems gefunden werden, so sind diese in einem Prototyping-Prozess ebenfalls viel leichter zu beseitigen als beim Test eines fertigen Systems. Durch die Einbeziehung der Nutzer in den Entwicklungsprozess, wie dies innerhalb

eines Rapide Prototyping vorgenommen werden kann, wird als weitere Nutzeffekt auch die Akzeptanz des Systems bei den Nutzern erhöht, weil auf ihre Anregungen schon während der Entwicklung eingegangen werden kann. Welcher Entwickler wäre aber dazu bereit ein bereits abgeschlossenes System nochmals vollständig zu verändern, weil der Benutzer andere Vorstellungen hat. Rapide Prototyping wurde auch aus diesem Grunde in der Entwicklung von KI-Systemen eingeführt. /Cawsey 03/
/Bachelier 04/ unterscheidet drei Arten des Rapide Prototyping (RP):

- Exploratives RP, als schrittweise Formulierung von Systemanforderungen,
- Experimentelles RP, in welchem nach und nach verschiedene Systemfunktionen formuliert und/oder überprüft werden und
- Evolutionäres RP, welches das System schrittweise von einer ersten Version mit kleiner überschaubarer Wissensbasis bis hin zum fertigen Ausbau entwickelt.

Betrachtet man nun das Rapide Prototyping rein als Integrationsvorgang, in welchem die Wissensbasis modelliert aber noch nicht vollständig abgebildet ist, dann wird man bei der Integration in eine XPS ohnehin evolutionär vorgehen müssen, weil sich die Wissensbasis ja nur nach und nach ins System überführen lässt. Verwendet man ein Entwicklungswerkzeug, also eine Shell oder ein Tool, so werden die beiden ersten Arten des RP obsolet, ansonsten gehört das explorative RP in die Phase des Systemdesigns und das experimentelle RP in die der Systemintegration. (siehe auch /Computerwoche 87/)
Die Integration eines XPS erfolgt also in verschiedenen Schritten, weil es immer wieder zu einer Abklärung mit den Benutzern kommen muss, allein um die Akzeptanz des Systems zu erreichen. XPS sind ganz erheblich von der Akzeptanz der Nutzer und der Organisation abhängig. Die Nutzer müssen, nach einer entsprechenden Schulung, das System auch anwenden wollen und daher von seiner Qualität überzeugt sein. Bei den hohen Kosten, die ein solches System verursacht, muss aber auch das Management von seiner Sinnhaftigkeit überzeugt sein. Solange das System eingesetzt wird muss es gewartet werde, was auch fortlaufende Kosten bedeutet und dies kann nur betriebswirtschaftlich gerechtfertigt werden, wenn das XPS einen Nutzen erzeugt. (siehe hierzu auch /Cawsey 03/)
Die während der Modellierung erstellten Dokumente (Wissensprotokolle, Modelle, Ontologien) können wie die Erkenntnisse, Wünsche und Anregungen

der Nutzer bei der schrittweise Einführung durch das Rapide Protyping sowohl als Meilensteinberichte wie als wieder verwendbare Dokumentation innerhalb der Wartung und Schulung verwendet werden. /Bachelier 04/

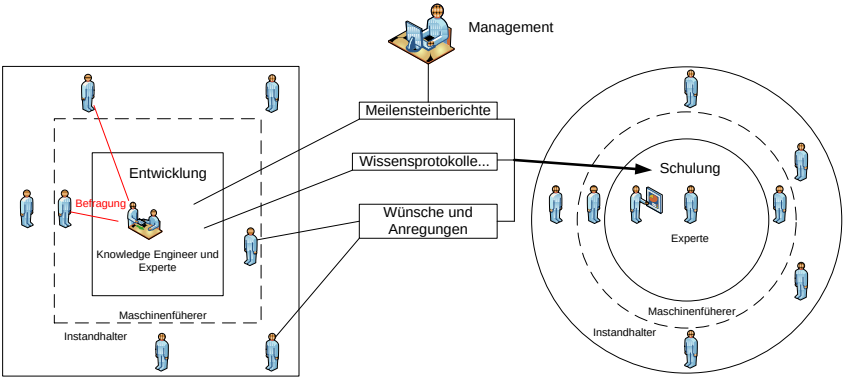


Abbildung 6.12: *Evolutionäre Umsetzung des Wissensmodells*

7 Schlussbemerkung und Ausblick

Für den Anfang der 90ziger Jahre des letzten Jahrhunderts waren von einer durch die Reagan Regierung in Auftrag gegebenen Markt-Studie ein Marktvolumen für die Expertensystem-Technologie von über einer Milliarde US-Dollar geschätzt worden. Diese zunächst für phantastisch angenommene Summe wurde weit übertroffen. Eine für die Bundesrepublik Deutschland vorgenommene Studie der Firma BrainwareGmbH sah bis ins Jahr 1995 ein Marktvolumen von 260 Millionen DM erreicht. /Pietzka 95/.

Die prognostizierte Verdopplung des Marktvolumens alle zwei Jahre erwies sich aber als viel zu optimistisch. So war zum Ende der 80ziger Jahre eine Phase hoher Entwicklungstätigkeiten feststellbar, die aber im Laufe der 90ziger Jahre abflaute. Dies wird in der Literatur auch durch das plötzliche Wiedererwachen der Forschungen zu neuronalen Netzen erklärt. /Bachelier 04/

In den vergangenen Dekaden habe ich immer wieder geradezu modehafte Erscheinungen bei den Forschungen zur KI bemerkt. Die Mode Expertensysteme, in der plötzlich jedes Programm, dass zwei Zahlen zusammenzählen konnte**,** als irgendwie gestaltetes wissensbasiertes System gedeutet wurde, wurde durch Multimedia, eLearning und Neuronale Netzen abgelöst. Immer wieder scheinen **s**ich die Kräfte der Forscher irgendwann zu erschöpfen, weil sie sich schlussendlich über die Komplexität der eigentlichen Aufgabe bewusst werden und sich einer neuen Mode zuwenden. Und so können auch so hoch angesehene Wissenschaftler wie Weizenbaum sich plötzlich von der KI abwenden, nachdem sie diese erst vehement gefördert haben. /Radermacher 93/

Hoffnung ergibt sich hier in den letzten Jahren durch die Entwicklung von hybriden Systemen, die verschiedene früher unversöhnliche Paradigmen wie WBS und neuronale Netze zu verbinden suchen. Vor allem im Bereich der Regelextraktion aus Netzen und der Erzeugung von Beispielmen**gen aus** Regeln, mit denen Netze trainiert werden können, werden zurzeit große Forschungsanstrengungen unternommen. /Bachelier 04/

Die Zukunft von Experten- und wissensbasierten Systemen wird meiner Meinung:

- nach zum einen zur Integration der Systeme in andere Systeme führen (siehe hierzu z. B. /Schmidt 04/)
    - etwa im Bereich der automatischen Diagnose, etwa im KFZ-Bereich, im Entwurf /Zores 98/ oder in der Robotik,

und

- zum anderen zur automatischen Aufbereitung der aus dem Problemfeld zugänglichen Daten mit Hilfe der Methoden des maschinellen Lernens (siehe hierzu etwa /Enk 99//Schleuder 94//Ewert 00//Khakzar 07/).

Das in diesem Buch vorgestellte XPS „Rotes Wunder“ wird in den nächsten Jahren sukzessive ausgebaut werden. Über den Fortgang der Entwicklung des Systems wird unter www.hs-merseburg/~hartmann informiert. Da die Entwicklung entweder auf der Sprache C oder dem frei zugänglichen Framework QFT beruht sind die Quellen frei zugänglich.
Bei der Recherche zu diesem Buch war ich doch erstaunt darüber, dass viele der Ansätze, die ich hier zu Papier brachte und die ich zusammen mit Kollegen bei der Entwicklung von für die Praxis tauglichen Expertensysteme fand, nicht in die wissenschaftliche Gedankenwelt Aufnahme fanden, obwohl ich schon vor fast 20 Jahren meine Karriere in der Industrie aufgab und obwohl es auch Veröffentlichungen zu diesem Thema gibt /Dröll 89/ und die Veröffentlichungen der INPRO. Mir scheinen viele Entwicklungen und Erkenntnisse in diesem Bereich noch in verschiedenen Schubladen zu liegen, was vielleicht auch an der Sprunghaftigkeit liegt, mit der viele Forscher im Bereich der KI gesegnet zu sein scheinen. Eine systematische Zusammenführung und Durchforschung der verschiedenen Bereiche aus Wissenschaft und Wirtschaft scheint hier angeraten zu sein.
Entwicklungen im Bereich der Künstlichen Intelligenz sind immer zeitaufwendig und teuer, unter anderen auch deshalb weil hier verschiedene Wissenschaften zusammengeführt werden müssen. Abgesehen von einigen sicher wichtigen öffentlich geförderten Forschungsleuchttürmen, wird die Weiterentwicklung ohne eine Zusammenarbeit zwischen Hochschulen und Industrie in Deutschland aber nicht möglich sein. Es liegt hier meiner Meinung nach aber vor allem an den Forschenden die Systeme den Vertretern der Wirtschaft geeignet vorzustellen und auch ihre Grenzen nicht zu verschweigen.
Egal wie man die zukünftigen Systeme auch immer nennen wird, die Erfahrungen von Experten zugänglich zu machen halte ich für alle Gesellschaften und Epochen für wichtig. Was anderes ist es denn, dass wir unseren Nachfahren unsere Erfahrungen in schriftlicher Form oder durch direkte Lehre zukommen lassen wollen. Die Methoden werden sich dabei im Laufe der Zeit vielleicht ändern, der Wille dass unsere Erfahrungen den

nachfolgenden Generationen in irgendeiner Weise helfen können ist eine Motivation die schon so alt wie die Menschheit selbst ist.

8 Literatur

/**Abecker 00**/ Abecker, Andreas: Software-Unterstützung für das betriebliche Wissensmanagement. Umwelt-Campus Birkenfeld 21. Juni 2000, Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI) GmbH, Kaiserslautern Forschungsgruppe Wissensmanagement, 2000.

/**Alpaydin 08**/ Alpaydin, Ethem: Maschinelles Lernen. Oldenbourg Wissenschaftsverlag GmbH, München, 2008.

/**Althoff 91**/ Althoff, Klaus-Dieter; Weß, Stefan: Fallbasiertes Problemlösen in Expertensystemen – begriffliche und inhaltliche Betrachtungen am Beispiel diagnostischer Aufgabenstellungen. Technischer Bericht. Universität Kaiserslautern, FB Informatik, Postfach 3049, 6750 Kaiserslautern. Althoff(wess)@informatik.uni-kl.de.

/**Angele 91**/ Angele,J.; Fensel, D.; Landes, D.; Studer, R.: KARL: An executable language for the Conceptual Model. In: Proceedings of the Knowledge Acquisition for Knowledge-Based Systems Workshop KAW'91, Banff, Canada, October 6-11, 1991.

/**Aue 90**/ Aue, Dirk; Baresch, Meike; Keller, Gerhard: URMEL Ein UnteRnehmens ModELlierungsansatz, Institut für Wirtschaftsinformatik, Universität des Saarlandes, 1990.

/**Bachelier 04**/ Bachelier, Günter: Wissensakquisition und Wissensrepräsentation in Expertensystemen. http://www.vi-anec.de/GBachelier/Talks/2004/XPS_Aqui.pdf, Zugriff 25.4.2010.

/**Barr 81-82**/ Barr, A.; Cohen, P.R.; Feigenbaum, E.A. (1982): the Handbook of Artificial Intelligence, Volume 1- 3 Addision Wesley Company, Inc, Reading Massachusetts.

/**Baumgartner 93**/ Baumgartner, P. : Der Hintergrund des Wissens. Vorarbeiten zu einer Kritik der programmierbaren Vernunft, Klagenfurt: Kärntner Druck- und Verlagsgesellschaft 1993.

/**Beierle 08**/ Beierle, Christoph; Kern-Isberner, Gabriele: Methoden wissensbasierter Systeme: Grundlagen, Algorithmen, Anwendungen, 4. Auflage. Vieweg+Teubner, GWV Fachverlage GmbH, Wiesbaden 2008.

/**Berlinews 08**/ http://www.berlinews.de/artikel.php?15139, Stichwort = „Intelligente Systeme“ , Zugriff 4.11.2010.

/**Bimazubute 05**/ Bimazubute, Raymond: Die Nachbereitung von Experteninterviews im expertenzentrierten Wissensmanagement, vorgelegte Doktorarbeit an der Technischen Fakultät der Universität Erlangen-Nürnberg zur Erlangung des Grades DOKTOR-INGENIEUR, 2005.

/**Bischoff 10**/ Bischoff, Rainer: Zur Geschichte von Rechentechnik und Datenverarbeitung. http://www.edgar-elsen.de/Historisches/Rechentechnik_und_DV_in_Jahreszahlen.htm . Zugriff 28.11.2010.

/**Blumstengel 98**/ Blumstengel: Mentale Modelle. http://dsor-fs.upb.de/~blumstengel/Mentale-Modelle.html, Zugriff 25.4.2010.

/Böhme 81/Böhme, G.: Einstieg in die mathematische Logik. Carl Hanser Verlag, München, Wien 1981.

/**Bohrer 04**/ Bohrer, Arndt: SaarCURA - Ein internetbasiertes Expertensystem zum Urheberrecht JurPC Web-Dok. 270/2004, Abs. 1 - 25 http://www.jurpc.de/aufsatz/20040270.htm, Zugriff 19.4.2010.

/**Brauer 91**/ Wilfried Brauer and Daniel Hernández:Verteilte Künstliche Intelligenz und kooperatives Arbeiten, 4. Internationaler GI-Kongress Wissensbasierte Systeme, München,23.-24. Oktober 1991, Proceedings, Informatik Fachberichte 291, Springer Verlag 1991.

/**Britannica 10**/ Encyclopaedia Britannica: Textsuche DENDRAL in Britannica Online Encyclopedia. www.britannica.com, Zugriff 19.4. 2010.

/**Britannica 10a**/ Encyclopaedia Britannica: Stichwort Mycin. http://www.britannica.com/ Zugriff 20.4.2010.

/**Brockhaus 10**/ Brockhaus multimedia 2010, wissenmedia GmbH, Güthersloh/München, 2010.

/**Brünner 02**/ Brünner, Gisela; Fiehler, Reinhard; Kindt, Walther (Hrsg.): Angewandte Diskursforschung, Band 2: Methoden und Anwendungsgebiete. Verlag für Gesprächsforschung , Radolfszell 2002.

/**Bruse 10**/ Bruse, Michael: Multi-Agenten Systeme. Geographisches Institut / Institut für Informatik Johannes Gutenberg-Universität Mainz. http://www.staff.uni-mainz.de/bruse/10Info_MAS/PDF/MAS_02KI.pdf . Zugriff 28.11.2010.

/**Budin 04**/ Budin, Gerhard: Multilinguale Lerncontententwicklung und –nutzung im Bereich der Ökologie – Erfahrungsbericht aus den Projekten Media Nova Naturae und Logos Gaias, 2004-09-28 Universität Hildesheim, Vortrag von Gerhard Budin, Universität Wien.

/**Buhl 96**/ Buhl, Hans-Ulrich; Roemer, Mark: Entwicklung und Anwendung dezentraler Problemlösungskompetenz in Finanzdienstleistungs-unternehmungen mit verteilten wissensbasierten Systemen in: Mayr, M.; et al., Hrsg., Beherrschung von Informationssystemen, Tagungsband der Informatik '96, Schriftenreihe der österreichischen Computer Gesellschaft, Klagenfurt, (Österreich), September 1996,Oldenbourg, Wien, 1996, S.143-144.

/**Cawsey 03**/ Cawsey, Alison: Künstliche Intelligenz im Klartext. Pearson Studium (Taschenbuch - 15. April 2003). ISBN 3-8273-7068-X. Pearson Education Deutschland GmbH, München 2003.

/**Christman-Jacoby 97**/ Christman-Jacoby, H; Maas, R.: Wissensmanagement im Projektumfeld auf Basis von Internet Technologien. In IM, 12. Jg. 1997, 16-26.

/**Computerwoche 84**/ Computerwoche 16.3.1984: "Expertensystem" löst Konfigurationsprobleme. www.Computerwoche.de. Zugriff 19.4.2010.

/**Computerwoche 87**/ Computerwoche 2.10.1987: Etablierte Methoden und Verfahren müssen adaptiert werden: Wissensakquisition legt Grundstein für Systemqualität. www.Computerwoche.de. Zugriff 19.4.2010.

/**Dabringer 09**/ Dabringer, Gerhard: Militärroboter. Einführung und ethische Fragestellungen. Materialien für Militärseelsorge. Institut für Religion und Frieden, Fasangartengasse 101, Objekt 7, 1130 Wien, 2009.

/**Dahlmann 06**/ Dahlmann, Sebastian: KnowFlow Report Engine – Ein graphen-basierter Ansatz zur automatischen Auswertung und Darstellung von Wissensprozessen einer Organisation Diplomarbeit, Institut für Wissensmanagement, Technische Universität Graz, Österreich., 2006.

/**De Kleer 86**/ de Kleer, J.: An Assumption Based TMS, AI-Journal 28, 127-162, 1986.

/**Dreyfus 79**/ Dreyfus, Hubert L.: What Computer can't do. The limits of Artificial Intelligence. Harper Colophone Books 1979.

/**Dreyfus 87**/ Dreyfus, H.; Dreyfus, S.: Künstliche Intelligenz. Von den Grenzen der Denkmaschine und dem Wert der Intuition, Reinbek b. Hamburg: Rowohlt 1987

/**Dröll 89**/ Dröll, M.; Hartmann, K.; Haupt, A. : Einsatz von Expertensystemen in der Fahrzeugindustrie. In: ZWF/CIM 11/89. Zeitschrift für wirtschaftliche Fertigung und Automatisierung. Carl Hanser Verlag, München, 1989, S. 607ff.

/**Duden 10**/ Office-Bibliothek: Version 5.06.0 (Build: 1072, 2.5.2009), Bibliographisches Institut AG, 1993-2009.

/**Eisfeld 01**/ Eisfeld, Michael: Wissensakquisition 10/03/2001 http://www-is.informatik.uni-oldenburg.de/~sauer/puk/config/akquisition.html, Zugriff 19.4.2010.

/**Enk 99**//Enk, Stefan: Expertensystem generalisiert trainierte Fallbeispiele zur Steuerung zeitvarianter Prozesse, http://www.uni-protokolle.de/nachrichten/id/47023/, Zugriff 30.5.2010 Institut für Elektrische Informationstechnik Tel. 0 53 23 72 38 52, Fax. 053 23 31 97, Leibnizstr. 28, 38678 Clausthal-Zellerfeld

/**Ewert 00**/ Ewert, Wolfgang; Leidholdt, Frank; Dürr, Holger: Repräsentation technologischer Wissensbasen für die Wissensakquisition durch Maschinelles Lernen in der generierenden Arbeitsplanerstellung Lehrstuhl für Fertigungslehre, Fakultät für Maschinenbau und Verfahrenstechnik, TU Chemnitz-Zwickau, PF 964 09009 Chemnitz {w.ewert, f.leidholdt, h.duerr}@mb2.tu-chemnitz.de.http://kluedo.ub.uni-l.de/volltexte/2000/175/ , Zugriff 19.4.2010.

/**Feigenbaum 84**/ Feigenbaum, Edward A.; McCorduck, Pamela: Die Fünfte Computer-Generation. Birkhäuser-Verlag Basel, Boston, Stuttgart,1984.
/Gabriel 08/ Gabriel, Roland: Expertensystem. http://www.enzyklopaedie-der-wirtschaftsinformatik.de/wi-enzyklopaedie/lexikon/technologien-methoden/KI-und-Softcomputing/Expertensystem . Zugriff 29.5.2010.

/**Gardner 89**/ Gardner, Howard: Dem Denken auf der Spur, Ernst Klett Verlag für Wissen und Bildung GmbH u. Co KG. Stuttgart, 1989.

/**Gens 90**/ Gens, M. : Möglichkeiten zur Darstellung des zeitlichen Verlaufs in prozeßorientierten und objektorientierten Wissensrepräsentationen. Unveröffentlichte Diplomarbeit der Technischen Fachhochschule Berlin. Berlin, Juli 1990.

/**Görz 03**/ Görz, G; Rollinger, C.-R.; Schneeberger, J. (Hrsg.): Handbuch der Künstlichen Intelligenz, 4. Auflage. Oldenbourg Wissenschaftsverlag GmbH, München 2003.

/**Gottlob 90**/ Gottlob, Georg; Frühwirt, Thomas; Horn, Werner: Expertensysteme. Springer Verlag Wien, 1990.

/**Gronau 05**/ Gronau, Norbert; Weber, Edzard: Analyse wissensintensiver Verwaltungsprozesse mit der Beschreibungssprache KMDL. In: Ralf Klischewski, Maria Wimmer (Hrsg.): Wissensbasiertes Prozessmanagement im E-Government. Münster: LIT, 2005.

/**Gronau 08**/ Gronau, Nobert: Modellierung. in wi-enzyklopaedie http://www.enzyklopaedie-der-wirtschaftsinformatik.de/wi-enzyklopaedie /lexikon/daten-wissen/Wissensmanagement/Wissensmodellierung/ index.html, Zugriff 1.12.2010.

/**Groschwitz 99**/ Groschwitz, Hans: Eine Diskussion der Problemkreise und Lösungswege bei der Realisierung eines dialoggesteuerten Expertensystems zur Auslegung primärgetakteter Schaltnetzteile. Vom Fachbereich Elektrotechnik der Bergischen Universität – Gesamthochschule Wuppertal angenommene Dissertation zur Erlangung des akademischen Grades eines Doktor-Ingenieurs.

/**Gusik 06**/ Gusik , Rainer: Wahrnehmen – ein Lehrbuch, http://eco.psy.ruhr-uni-bochum.de/download/Guski-Lehrbuch/. Zugriff 28.4.2010. KohlhammerVerlag, Stuttgart 2006.

/**Habel 85**/ Habel, C; Rollinger, C.-R. (1985): Lernen und Wissensakquisition. In: Habel, C. (Hrsg.): Künstliche Intelligenz. Repräsentation von Wissen und natürlichsprachliche Systeme. Frühjahrsschule Dassel (Sollingen), März 1984. Springer Verlag, Berlin, Heidelberg, New York, Tokio, 1985, S. 249ff..

/**Hartmann 84**/ Hartmann, K. : Aufbau einer künstlichen Welt geometrischer Objekte für das CAD-System Baustein GEOMETRIE. Unveröffentlichte Studienarbeit der Technischen Universität Berlin, Fachbereich Informatik.

/**Hartmann 85**/ Hartmann, K.: Konzept eines Expertensystems zur Prozessdiagnose. Unveröffentlichtes Konzept der Firma INPRO, Nürnbergerstr. 68/69, 1000 Berlin 10, 1985.

/**Hartmann 92**/ Hartmann, K.: Aufbau eines wissensbasierten Systems zur Kontrolle von lokalen Netzwerken. Zur Erlangen des akademischen Grades Doktor-Ingenieur vorgelegte Dissertation. Technische Universität Berlin, 1992.

/**Hartmann 01**/ Hartmann, Karsten: Geschichte der Künstlichen Intelligenz. Versuch einer Standortbestimmung. in: Nühlen, Maria (Hrsg.): Merseburger Ringvorlesung, Geschichte und Geschichten, Fachhochschule Merseburg, 2001.

/**Hartmann 09**/ Hartmann, Karsten: Einführung in die Praxis der Wissensdarstellung. Neverendingland GmbH 2009.

/Hasebrook 95/ Hasebrook, J.: Multimedia-Psychologie: Eine neue Perspektive menschlicher Kommunikation. Spektrum Verlag; Heidelberg, Berlin, Oxford; 1995

/**Heinsohn 99**/ Heinsohn, Jochen; Socher-Ambrosius, Rolf: Wissensverarbeitung. Eine Einführung; Spektrum, Akademischer Verlag Heidelberg [u.a.], 1999.

/Herczeg 05/ Herczeg, Michael: Software-Ergonomie: Grundlagen der Mensch-Computer-Kommunikation, Oldenbourg Wissenschaftsverlag GmbH, München 2005.

/**Herczeg 06**/ Herczeg, Michael: Differenzierung mentaler und konzeptueller Modelle und ihrer Abbildungen als Grundlage für das Cognitive System Engineering. In Grandt, M.: Cognitive Systems Engineering in der Fahrzeug- und Prozessführung. DGLR- Bericht 2006-02.

/**Heß 02**/ Heß, Lena-Luisa, Expertensysteme in der Medizin, 26. 11. 2002. http://members.tripod.com/lena_hess/data/skripte/Expertensysteme.pdf, Zugriff 20.4.2010.

/**Hinterding 95**/ Hinterding, Ursula: Einsatz von Expertensystemen bei der Entscheidung zwischen Leasing und Kreditkauf, http://www.wiso.uni-koeln.de/leasing/publikationen_/mub/nr_19/hinterdi.pdf, Zugriff 1.6.2010, in: MuB Leasing, 10. Jg., 1995, Nr. 19, S. 55-90.

/**Hubig 04**/ Hubig Christoph: Wissensmanagement und Kommunikation in der E-Economy (BW) Exzerpte aus Christoph Hubig: Wissensmanagement und Kommunikation in der E-Economy. Zum Widerspruch zwischen Rationalisierung und Kompetenzerweiterung. In: U. Frank (Hrsg.): Wissenschaftstheorie in Ökonomie und Wirtschaftsinformatik. Wiesbaden 2004. S. 211ff. http://philo.at/wiki/index.php/Christoph_Hubig:_Wissensmanagment_und_Kommunikation_in_der_E-Economy_(BW)

/**INPRO 88**/ Innovationsgesellschaft für fortgeschrittene Produktionssysteme in der Fahrzeugindustrie: PRIMUS, Beschreibung einer Expertensystemshell. Prospekt zum internen Gebrauch. INPRO, Nürnbergerstr. 68/69, 1000 Berlin 10, 1988.

/**INPRO 89**/ Innovationsgesellschaft für fortgeschrittene Produktionssysteme in der Fahrzeugindustrie: IXRO - Ein Expertensystem zur Fehlerdiagnose von Robotern, Abschlußbericht, Dezember 1989. Zum internen Gebrauch. INPRO, Nürnbergerstr. 68/69, 1000 Berlin 10, 1989.

/**INPRO 90**/ Innovationsgesellschaft für fortgeschrittene Produktionssysteme in der Fahrzeugindustrie: IXTRA - Ein Expertensystem zur Transferstraßen Fehlerdiagnose, Abschlußbericht, Mai 1990. Zum internen Gebrauch. INPRO, Nürnbergerstr. 68/69, 1000 Berlin 10, 1990.

/**INPRO 93**/ Innovationsgesellschaft für fortgeschrittene Produktionssysteme in der Fahrzeugindustrie: SIGMA - Eine Expertensystemshell zur technischen Diagnose, Abschlußbericht, 1993. Zum internen Gebrauch. INPRO, Nürnbergerstr. 68/69, 1000 Berlin 10, 1990.

/**Jarz 97**/ Jarz, Ewald M.: Entwicklung multimedialer Systeme. Planung von Lern- und Masseninformationssystemen. Mit einem Geleitwort von Friedrich Roithmayr, Wiesbaden: Gabler-Verlag, Deutscher Universitäts-Verlag 1997, 361 S.

/**Juršič 03**/ Juršič, Nikolai: Wissensmodellierung Diplomarbeit, Fakultät für Human- und Sozialwissenschaften der Universität Wien, 2003.

/**Khakzar 07**/ Khakzar, Karim: Interaktives Expertensystem für Maßkonfektion, Hochschule Fulda Fachbereich Angewandte Informatik. http://www.intexma.info/index.php?show= pspiegel, Zugriff 25.4.2010.

/**Kleinhans 89**/ Kleinhans, A. M.: Wissensverarbeitung im Management, Verlag Peter Lang, Frankfurt am Main u.a., 1989.

/**Knemeyer 94**/ Knemeyer, Ullrich: Expertensysteme in der Assekuranz. Chancen und Risiken ihrer Realisierung, dargestellt am Beispiel des ökologischen Risk Managements in der Landwirtschaft, Verlag Versicherungswirtschaft e.V., Karlsruhe 1994.

/**Knill 03**/Marcus Knill: Interviewtechnik, **http://www.rhetorik.ch/Interviewtechnik/Interviewtechnik. html**, Zugriff 25.4.2010.

/**Köchert 10**/ Expertensysteme im Hypertextzeitalter. Tenea-Verlag Berlin. http://www.finexs.de/inhalt/buch.html. Zugriff 29.5.2010.

/Kunze 03/ Kunze, Manuela; Kapfer, Jörg: Werkzeuge zur Wissensakquisition und Informationsverdichtung aus Dokumenten http://wdok.cs.uni-magdeburg.de/forschung/projekte/ infofusion/ Zugriff 25.4.2010.

/**Kurbel 92**/ Kurbel, Karl: Entwicklung und Einsatz von Expertensystemen, Eine anwendungs-orientierte Einführung in wissensbasierte Systeme. 2. Auflage. Springer Verlag, Berlin Heidelberg New York Tokyo 1992.

/**Lahr 09**/ Lahres, Bernhard; Raýman, Gregor: Objektorientierte Programmierung, Galileo Press, Bonn 2009.

/**Langenscheid 97**/ Langenscheid Lilliput Englisch-Deutsch, Langenscheid KG, Berlin und München, 1997.

/**Lawson 97**/ Lawson, Gavin; Lukose, Dickson : Rapid prototyping of executable problem solving methods using MODEL-ECS in: Abdul Sattar (Ed.): Advanced Topics in Artificial Intelligence 10th Australian Joint Conference on Artificial Intelligence, AI'97 Perth, Australia, November 30 – December 4, 1997 Proceedings, Springer Verlag Berlin, Heidelberg, 1997.

/**Lederberg 87**/ Lederberg, Joshua: How DENDRAL was conceived and born. Rockefeller University New York, N.Y. . ACM Symposium on the History of Medical Informatics National Library of Medicine November 5, 1987.

/**Lehner 09**/ Lehner, Franz: Wissensmanagement. Grundlagen, Methoden und technische Unterstützung. 3. Auflage. Carl Hanser Verlag München, Wien, 2009.

/**Lenz 07**/ Lenz, Karl: Methoden der empirischen Sozialforschung III. Komplex: Qualitative Forschungsmethoden, Vorlesung im Wintersemester 2007/08 www.tu-dresden.de/phfis/lenz, Zugriff April 2010.

/**lexikon stangel 10**/ Lexikon für Psychologie und Pädagogik, http://lexikon.stangl.eu/32/ operationalisierung/. Zugriff 25.4.2010.

/**Lutz 02**/ Lutz, Michael; Möltgen, Jörn; Kuhn, Werner: Ontologien zur Spezifikation von Informationssystemen für Verkehrsplaner. Dipl.-Landsch.-Ökol. Michael Lutz, Uni Münster, Institut für Geoinformatik, Robert-Koch-Str. 26-28, 48149 Münster, lutzm@ifgi.uni-muenster.de; Dipl.-Geogr. Jörn Möltgen, Uni Münster, Institut für Geoinformatik, Robert-Koch-Str. 26-28, 48149 Münster, moltgej@ifgi.uni-muenster.de; Prof. Dr. Werner Kuhn, Uni Münster, Institut für Geoinformatik, Robert-Koch-Str. 26-28, 48149 Münster, kuhn@ifgi.uni-muenster.de. In: MULTIMEDIAPLAN.AT & IEMAR, TU Wien, S.151-156, 2002.

/**Marchand 85**/ Marchand, H. (1985b): Objektorientierte Wissensdarstellung in industriellen Expertensystemen. In: Brauer, W.; Radig, B.: Wissensbasierte Systeme, Gl-Kongreß 1985. Springer Verlag, Berlin, Heidelberg, 1985, S. 145ff.

/**Maxima 09**/ Maxima, a Computer Algebra System. http://maxima.sourceforge.net/ Zugriff 25.4.2010.

/Mc Corduck 79/ Mc Corduck, P.: Machine who think. A personal Inquiery into the History and Prospects of Artificial Intelligence. W.H. Freeman and Company, San Francisco 1979.

/**Michalski 83**/ Michalski, R.S.; Carboneil, J.G.; Mitchell, T.M, (1983): Machine Learning. An Artificial Intelligence Approach. Tioga Publishing Company, Palo Alto, 1983.

/**Michie 94**/ Michie, D.; Spiegelhalter, D. J.: Machine Learning, Neural and Statistical Classification. In: Ellis Horwood Series in Artificial Intelligence. E. Horwood Verlag, New York 1994.

/**Mitchel 97**/ Mitchell, Thomas: *Machine Learning*. Mcgraw-Hill, London 1997.

/**Morik 05**/ Morik, Katharina; Boulicaut, Jean-Francois; Siebes, Arno (Eds): International Seminar - Local Pattern Detection, Dagstuhl Castle, Germany, April 2004. Springer Verlag, Berlin, Heidelberg, New York 2005.

/**Morik 93**/ Morik, Katharina: Knowledge acquisition and machine learning: theory, methods, and applications, Academic Press, 1993.

/**Mörler 84**/ Mörler, Gerhard: Expertensystem - nur ein neues Modewort? in Computerwoche 31.8.1984. Computerwoche.de. Zugriff 19.4.2010.

/Parikh 89/ Parikh, B. : Konzeption und Implementierung eines graphischen Editors für Aufgabennetze unter GEM. Unveröffentlichte Diplomarbeit. Institut für Angewandte Informatik der Technischen Universität Berlin. Berlin, Juni 1989.

/**Petcu 98**/ Petcu, Cornelia: Expertensysteme. http://www.jurpc.de/aufsatz/19980021.htm . Zugriff 29.5.2010.

/**Petti 03**/: Petti, Richard: XML, why Mma is popular, was...Re: [Maxima] Lunch with George Carrette. http://www.math.utexas.edu/pipermail/maxima/2003/005861.html, Zugriff 25.4.2010.

/**Pfeffer 02**/ Pfeffer, Sabine: Expertensysteme in der Medizin, JurPC Web-Dok. 91/202, Abs. 1-23. http://www.jurpc.de/ Zugriff 20.4.2010.

/**Pietzka 95**/ Pietzka, Lutz: Techniken und Methoden wissensbasierter Systeme: Expertensysteme in Bibliothek, Information und Dokumentation. Bibliothek 19. 1995. Nr.3 Pietzka – Techniken und Methoden wissensbasierter Systeme. S. 371- 385. http://www.bibliothek-saur.de/1995_3/371-385.pdf. Zugriff 20.4.2010.

/**Porr 02**/ Porr, Bernd: Systemtheorie und Naturwissenschaft: eine interdisziplinäre Analyse von Niklas Luhmanns Werk. Deutscher Universitätsverlag GmbH, Wiesbaden 2002.

/**Puppe 88**/ Puppe, F.: Einführung in Expertensysteme. Studienreihe Informatik. Springer-Verlag, Berlin, Heidelberg, New York, 1988.

/**Puppe 91**/ Puppe, Frank: Einführung in Expertensysteme; Springer Verlag Berlin Heidelberg, 1991.

/**Radermacher 93**/ Radermacher, Josef: Für einen Einsatz von XPS muss die Infrastruktur stimmen Fehlentwicklungen beruhen oft auf hausgemachten Problemen. in Computerwoche 22.1.1993. Computerwoche.de. Zugriff 6.5.2010.

/**Rauch 82**/ Rauch, Wolf: Büro-Informations-Systeme. Sozialwissenschaftliche Aspekte der Büro-Automatisierung durch Informations-Systeme; Verlag Böhlau Wien, Graz [u.a.] 1982.

/**Rehberger 95**/ Rehberger, Dirk: Das System DENDRAL.Ausarbeitung zu einem Seminarvortrag im Seminar „Wissensbasierte Systeme", Prof. Dr. M. Schmidt-Schauß, Wintersemester 1995/96,Fachbereich Informatik, JWG-Universität Frankfurt am Main. www.ki.informatik.uni-frankfurt.de/lehre/case/dendral.ps. Zugriff 28.11.2010.

/**Reichertz 94**/ Reichertz, Jo: Polizeiliche Expertensysteme: Illusion oder Verheißung? In: R. Hitzler & A. Honer & Ch. Maeder (Hrsg.) Expertenwissen. Opladen. S. 193 - 213.

/**Reif 00**/ Reif, Gerald: *Moderne Aspekte des Wissensverarbeitung. Ein interaktiver Lernbehelf für das Web Based Training*. Diplomarbeit *an der* Technischen Universität Graz, Institut für Informationsverarbeitung und Computergestützte neue Medien (IICM), Technische Universität Graz A-8010 Graz, Jänner 2000.

/**Reinmann-Rothmeier 97**/ Reinmann-Rothmeier; G., Mandl, H.: Wissensmanagement: Phänomene-Analyse-Forschung-Bildung. In: Ludwig-Maximilian Universität, Lehrstuhl für Empirische Pädagogik und Pädagogische Psychologie, 83, 1997.

/**Reminger 88**/ Reminger, Brigitte: Die technischen Hürden sind noch lange nicht überwunden: Expertensysteme lassen viele Wünsche offen. in: Computerwoche 16.9.1988. Computerwoche.de. Zugriff 6.5.2010.

/**Remus 02**/ Remus, Ulrich: Prozeßorientiertes Wissensmanagement. Konzepte und Modellierung. Dissertation zur Erlang des akademischen Grades eines Doktors der Wirtschaftswissenschaften. Wirtschaftswissenschaftliche Fakultät der Universität Regensburg, 2002.

/**Richter 10**/ Richter, Jürgen: Vorlesung zu Wissensbasierten Systeme. Fachhochschule Südwestfalen, Hochschule für Technik und Wirtschaft. http://www3.fh-swf.de/fbei/richter_j/download/Kapitel-1.pdf, Zugriff 25.4.2010.

/**Rieger 04**/ Rieger, Peter; Heymann, Stephan; Müller, Heiko: Datenbankgestützte Wissensakquisition in den Lebenswissenschaften Datenbankspektrum 10/2004. http://www.dbis.informatik.hu-berlin.de/fileadmin /research /papers /journals/2004-dbspektrum-rieger.pdf.

/**Ryle 10**/ Ryle; Baumgartner; u.a.: Philosophische Ansätze zur Wissensklassifikation. http://www.teachsam.de/psy/psy_kog/lernth/wiss/wiss_2_1.htm, Zugriff 25.4.2010.

/**Scheer 90**/ Scheer, A.-W.: CIM – Der computergesteuerte Industriebetrieb. Springer-Verlag, New York, Berlin, Heidelberg, Tokyo 1990.

/**Schleuder 94**/ Schleuder, Martin: Expertensysteme und neuronale Netze im intelligenten Wohnhaus der Zukunft, Hochschulschriften Band 15, Metropolis Verlag für Ökonomie, Gesellschaft und Politik, 1994.

/**Schlieder 01**/ Schlieder, Christopher; Visser, Ubbo: Vorlesung Künstliche Intelligenz II, Sommersemester 2001, Universität Bremen.

/**Schmidt 04**/ Schmidt, Thorsten; Rosner, Hans-Joachim: Ausgewählte Methoden der Künstlichen Intelligenz in der Geographie, Hallesches Jahrbuch Geowissenschaften, Bd. 26. Halle (Saale) 2004.

/**Schmidt 10**/ Schmidt, Thorsten; Fuchs, David: A-Stern Algorithmus. Universität Tübingen, Geographisches Institut http://www.geosimulation.de/methoden/a_stern_algorithmus.htm, Zugriff 19.4.2010.

/**Schneider 10**/ Schneider, Oliver: Grenzen im Einsatz von Expertensystemen Fachpräsentation. http://www.oliver-koelzow.de/fileadmin/dokumente/ Expertensysteme.ppt. Zugriff 20.4.2010.

/**Spreckelsen 08**/ Spreckelsen, Cord; Spitzer, Klaus: Wissensbasen und Expertensysteme in der Medizin. Vieweg+Teubner, GWV Fachverlage GmbH, Wiesbaden 2008.

/**Spur 94**/ Spur, G.; Stöferle, TH.: Handbuch der Fertigungstechnik, Band 6 Fabrikbetrieb. Carl Hanser Verlag, München, Wien, 1994.

/**Stanford 10**/ hci.stanford.edu/winograd/shrdlu/ . Zugriff 29.5.2010.

/**Steinmann 93**/ Steinmann, Dieter: Einsatzmöglichkeiten von Expertensystemen in integrierten Systemen der Produktionsplanung und -steuerung (PPS), Physica Verlag,1993.

/**Steinmüller 05**/ Steinmüller, Johannes: Expertensysteme Vorlesung an der TU Chemnitz Sommersemester 2005, stj@informatik.tu-chemnitz.de, http://www.tu-chemnitz.de/~stj/stj.html, Zugriff 1.6.2010.

/**Strube 89**/ Strube, G. (1989): Episodisches Wissen. Arbeitspapiere der GMD, 385, 10-26

/**Turing 87**/ Dotzler, B; Kittler, F (Hrsg.): Intelligence Service. Schriften. Brinkmann & Bosse, 1987.

/**Uni Würzburg 04**/ Entscheidungsunterstützende Programme. http://ki.informatik.uni-wuerzburg.de/teach/ws-2004-2005/medInfo/documents/MedInfo-2004_05-09.pdf. Zugriff 25.4.2010.

/**Unikassel 07**/ http://www.kde.cs.uni-kassel.de/lehre/ws2007-8/KI/folien/4_01_Methoden_und_Techniken.pdf, Zugriff 25.11.2010.

/**Universität Würzburg 03**/ Skript zur Vorlesung WMS-WAQ.http://ki.informatik.uni-wuerzburg.de/teach/ws-2002-2003/wms/uebungen/skript/WMS-WAQ.4.pdf, Zugriff 25.4.2010.

/**Veloso 89**/ Veloso, M. M. & Carbonell, J. G.: Learning Analogies by Analogy - The Closed Loop of Memory Organization and Problem Solving. In: Hammond, K. (ed.) (1989a). Proc. of the 2nd DARPA Workshop on Case-Based Reasoning. Holliday Inn, Pensacola Beach: Morgan Kaufmann

/**Wachsmuth 94**/ Wachsmuth, Ipke; Meyer-Fujara, Josef: Wissensbasierte Informationsverarbeitung mit Expertensystemen: Wissen – Fachwissen – Erfahrungswissen in: H. Best, B. Endres-Niggemeyer, M. Herfurth & H.P.Ohly (Eds.): Informations- und Wissensverarbeitung in den Sozialwissenschaften (pp. 103-113). Opladen: Westdeutscher Verlag, 1994.

/**Wagner 08**/ Wagner, Thomas: Agentenunterstütztes Engineering von Automatisierungsanlagen. Vorgelegte Dissertation, Fakultät Informatik, Elektrotechnik und Informationstechnik, Universität Stuttgart 2008.

/**Wagner 93**/ Wagner, Peter: Wissensakquisition für Expertensysteme, Universität Halle, Zeitschrift für Agrarinformatik, H. 1, 1. Jg. 1993, S.10-18. http://lb.landw.uni-halle.de/publikationen/veroe42/veroe42.htm, Zugriff 25.4.2010.

/**Walde 38**/ Walde, Alois: Lateinisches etymologisches Wörterbuch, 3. Aufl. Heidelberg 1938, II, S. 784f.

/**Waves 02**/ Wissensakquisition für ein taxonomisches Modell und Entwicklung einer Bewertungskomponente zur Unterstützung von WAVES, Probleme der Kommunikation, oder inwieweit ist Interdisziplinarität machbar? http://www.usf.uni-kassel.de/waves/deutsch/endbericht/ Zugriff 25.4.2010.

/**Wersig 71**/: Wersig, Gernot: Information - Kommunikation - Dokumentation. Ein Beitrag zur Orientierung der Informations- und Dokumentationswissenschaft*;* Verl. Dokumentation München-Pullach [u.a.], 1971.

/**Wiegand 04**/ Wiegand, Sven: Knowledge Engineering des Managements der Informationssystemsicherheit, Entwicklung einer Diagnose-Shell zur Unterstützung von Informationssystemsicherheit. Vorgelegte Dissertation, Fachbereich Wirtschaftswissenschaften der Universität Duisburg-Essen, Standort Essen, 2004.

/**wiki 10**/ Wikipedia http://en.wikipedia.org/wiki/Mental_operations, Suchbegriff: Mental operations, Zugriff 25.4.2010.

/**wiki 10a**/ http://de.wikipedia.org/wiki/Alpha-Beta-Suche, Suchbegriff: Alpha-Beta-Suche, Zugriff: 25.4.2010.

/**wiki 10b**/ http://de.wikipedia.org/wiki/Experte, Suchbegriff: Experte, Zugriff: 25.4.2010.

/**wiki 10c**/ http://de.wikipedia.org/wiki/Mentales_Modell, Suchbegriff: Mentales Modell, Zugriff: 25.4.2010.

/**wiki 10d**/ http://de.wikipedia.org/wiki/_Operationalisierung , Suchbegriff: Operationalisierung, Zugriff: 25.4.2010.

/**wiki 10e**/ http://de.wikipedia.org/wiki/_Resolution , Suchbegriff: Resolution, Zugriff: 25.4.2010.

/**wiki 10f**/ http://de.wikipedia.org/wiki/_Wissen , Suchbegriff: Wissen, Zugriff: 25.4.2010.

/**wiki 10g**/ http://de.wikipedia.org/w/index.php?title=Datei:Ki-methoden.png, Zugriff 8.7.2010.

/**wiki 10h**/ http://de.wikipedia.org/wiki/Diagnose , Zugriff 27.10.2010.

/**wiki 10i**/ http://de.wikipedia.org/wiki/ Stichwort=Xcon , Zugriff 4.11.2010.

/**wiki 10k**/ http://de.wikipedia.org/wiki/Maschinelles_Lernen, Zugriff 9.12.2010.

/**wiki 10l**/ http://de.wikipedia.org/wiki/Datalog, Zugriff 9.12.2010.

/**wikigps 10**/ General Problem Solver. http://de.wikipedia.org/w/index.php?oldid=64624607. Zugriff 25.4.2010.

/**Willke 96**/ Willke, H.: Dimensionen des Wissensmanagements – Zum Zusammenhang von gesellschaftlicher und organisatorischer Wissensbasierung. In: Schreyögg, G., Conrad, P.: Managementforschung, Band 6: Wissensmanagement, de Gruyter, Berlin, New York, 1996, 263-304.

/**Wischnewsky 10**/ Wischnewsky, M.B.: Expertensysteme in der Medizin zur Unterstützung von Diagnose und Therapie. Zentrum für Multimedia in der Lehre (ZMML) und Technologiezentrum Informatik, Universität Bremen. www.d-lecture.de/projekt/elearning_wisch/Expertensysteme_Unterstuetzung.PDF. Zugriff 20.4.2010.

/**Wolff 05**/ Wolff, Christian: Generierung ontologischer Konzepte und Relationen durch Text Mining-Verfahren." In: Knowledge eXtended. Die Zusammenarbeit von Wissenschaftlern, Bibliothekaren und IT-Spezialisten. Hrsg. v. Forschungszentrum Jülich. Jülich: FZ Jülich. Schriften des Forschungszentrums Jülich, Reihe Bibliothek, Band 14, 155-162. http://www.fz-juelich.de/zb/datapool/page/768/Wolff_Abstract.pdf, Zugriff 25.4.2010.

/**Wollny 03**/ Wollny, Stefan: Erklärungsfähigkeit kooperierender regelbasierter Expertensysteme zum diagnostischen Problemlösen. Von der Fakultät IV – Elektrotechnik und Informatik der Technischen Universität Berlin zur Erlangung des Grades Doktor der Ingenieurwissenschaften, Berlin, 2003.

/**Zabel 00**/ Zabel, Frank; Hempel, Tino: Expertensysteme, Seminar zur Didaktik der Informatik. Greifswald 2000.

/**Zores 98**/ Zores, Robert: Integrierte Entwurfsmethoden als Basis eines Expertensystems für transsonischen Profilentwurf. Vorgelegte Dissertation, Georg-August-Universität zu Göttingen. Göttingen 1998.

Anhang A: Bauteilebaum der Transferstraße „Rotes Wunder“

Rotes_Wunder besteht aus:

Foerderband_mit_2_Pushern

Foerderband

- FB_Motor (Asset_ID => 7M1)
- Foerderkette1 (Asset_ID => FK1)
- Foerderkette2 (Asset_ID => FK2)
- Getriebe (Asset_ID => G7M1)
- Umsetzer (Asset_ID => U7M1)

Pusher_links

- BPL_Sensor_Pusher_vorn (Asset_ID => 3S1)
- BPL_Sensor_Pusher_hinten (Asset_ID => 3S2)
- BPL_Sensor_Pusher_belegt (Asset_ID => 3S3)
- BPL_Motor (Asset_ID => 7M2)
- BPL_Schlitten (Asset_ID => 3U1)
- BPL_Schlittenfuehrung (Asset_ID => 3U2)
- BPL_Stoessel (Asset_ID => 3U3)

Pusher_rechts

- BPR_Sensor_Pusher_vorn (Asset_ID => 3S4)
- BPR_Sensor_Pusher_hinten (Asset_ID => 3S5)
- BPR_Sensor_Pusher_belegt (Asset_ID => 3S6)
- BPR_Motor (Asset_ID => 7M3)
- BPR_Schlitten (Asset_ID => 3U4)
- BPR_Schlittenfuehrung (Asset_ID => 3U5)
- BPR_Stoessel (Asset_ID => 3U6)

Ablageplatz_links

- BAL_Bedientaster (Asset_ID => 4S1)
- BAL_Sensor_belegt (Asset_ID => 3S7)
- BAL_Led_belegt (Asset_ID => 5H1)

Ablageplatz_rechts

- BAR_Bedientaster (Asset_ID => 4S2)
- BAR_Sensor_belegt (Asset_ID => 3S8)
- BAR_Led_belegt (Asset_ID => 5H2)

Umsetzer_mit_Sauggreifer

Foerderband_Links
Motor_Links(Asset_ID => 7M3)
InduktiverNaeherungsschalter_Links(Asset_ID => 3S6)
Taster_Notaus_Links (Asset_ID => 3S8)
Foerderband_Rechts
Motor_Rechts(Asset_ID => 7M4)
InduktiverNaeherungsschalter_Rechts (Asset_ID => 3S7)
Taster_Notaus_Rechts (Asset_ID => 4S1)
Zwischenlager
Reedkontakt (Asset_ID => 4S2)
Horizontaler_Arm
Motor_Horizontal (Asset_ID => 7M1)
Taster_Horizontal_Links (Asset_ID => 3S3)
Taster_Horizontal_Mitte (Asset_ID => 3S2)
Taster_Horizontal_Rechts (Asset_ID =>3S1)
Inkrementierer (Asset_ID =>__)
Vertikaler_Arm
Motor_Vertikal (Asset_ID => 7M2)
Taster_Bewegung (Asset_ID => 3S4)
Taster_Erfassung(Asset_ID => 3S5)
Greifer(Asset_ID =>__0)
Ventile (Asset_ID =>__1)
Pumpe
EMagnet (Asset_ID =>__2)

Fraese_Gesamtanlage

Fraese_Foerderband
Fraese_Foerderband_Antrieb
Fraese_Foerderband_Antrieb_Motor
Fraese_Foerderband_Antrieb_Kette
Fraese_Foerderband_Antrieb_Getriebe
Fraese_Foerderband_Kette
Fraese_Foerderband_Antrieb_Kette_Kettengliedern
Fraese_Foerderband_Getriebe
Fraese_Foerderband_Antrieb_Getriebe_Schneckengewinde
Fraese_Foerderband_Antrieb_Getriebe_Zahnrad
Fraese_Foerderband_Antrieb_Getriebe_Welle
Fraese_Werkzeugmaschine
Fraese_Werkzeugmaschine_Positionierungseinheit
Fraese_Werkzeugmaschine_Positionierungseinheit_Antrieb_Z

Fraese_Werkzeugmaschine_Positionierungseinheit_Antrieb_Z_Motor
Fraese_Werkzeugmaschine_Positionierungseinheit_Antrieb_Z_Getriebe
Fraese_Werkzeugmaschine_Positionierungseinheit_Antrieb_Z_Anschlusskabel_lila_grau
Fraese_Werkzeugmaschine_Positionierungseinheit_Antrieb_Y
Fraese_Werkzeugmaschine_Positionierungseinheit_Antrieb_Y_Motor
Fraese_Werkzeugmaschine_Positionierungseinheit_Antrieb_Y_Getriebe
Fraese_Werkzeugmaschine_Positionierungseinheit_Antrieb_Y_Anschlusskabel_schwarz_weiss
Fraese_Werkzeugmaschine_Positionierungseinheit_SchalterZp
Fraese_Werkzeugmaschine_Positionierungseinheit_SchalterZp_Anschlusskabel_gruen_ blau
Fraese_Werkzeugmaschine_Positionierungseinheit_SchalterZm
Fraese_Werkzeugmaschine_Positionierungseinheit_SchalterZm_Anschlusskabel_gelb_ blau
Fraese_Werkzeugmaschine_Positionierungseinheit_SchalterYp
Fraese_Werkzeugmaschine_Positionierungseinheit_SchalterYp_Anschlusskabel_braun_ orange
Fraese_Werkzeugmaschine_Positionierungseinheit_SchalterYm
Fraese_Werkzeugmaschine_Positionierungseinheit_SchalterYm_Anschlusskabel_unbekannt_unbekannt
Fraese_Werkzeugmaschine_Fraeseinheit
Fraese_Werkzeugmaschine_Fraeseinheit_Drehmotor
Fraese_Werkzeugmaschine_Fraeseinheit_Drehmotor_Anschlusskabel_gelb_orange
Fraese_Werkzeugmaschine_Fraeseinheit_Fraeskopf

Foerderband_Drehtisch_Rollenbahn

Drehtisch
Abdeckplatte (),
Foerderung
Antrieb_Foerderung
Motor_Antrieb_Foerderung(Asset_ID => 7M1)
Ritzel_Antrieb_Foerderung
Schneckengewinde_Antrieb_Foerderung
Getriebe_Foerderung

- Getriebe_Foerderung_Sekundaerachse
- Getriebe_Foerderung_Antriebsachse
- Getriebe_Foerderung_Ritzel_AAL
- Getriebe_Foerderung_Ritzel_AAR
- Getriebe_Foerderung_Ritzel_SAL
- Getriebe_Foerderung_Ritzel_SAR
- Getriebe_Foerderung_Kette_links
  - Getriebe_Foerderung_Kette_links_Kettenglied
- Getriebe_Foerderung_Kette_rechts
  - Getriebe_Foerderung_Kette_rechts_Kettenglied

Drehung

- Antrieb_Drehung
  - Motor_Antrieb_Drehung (Asset_ID => 7M2),
    Ritzel_Antrieb_Drehung
- Getriebe_Drehung
  - Getriebe_Drehung_Ritzel_klein
  - Getriebe_Drehung_Ritzel_gross
  - Getriebe_Drehung_Ritzel_flach

SCH_AS (Asset_ID => 3S1)

SCH_LS

INS_Drehtisch (Asset_ID => 3S3)

Foerderband_kurz

- Antrieb_kurz
  - Motor_Foerderband_kurz (Asset_ID => 7M3),
    Schneckengewinde_Foerderband_kurz
- Getriebe_kurz
  - Getriebe_Foerderband_kurz_Ritzel_Antrieb
  - Getriebe_Foerderband_kurz_Sekundaerachse
  - Getriebe_Foerderband_kurz_Antriebsachse
  - Getriebe_Foerderband_kurz_Ritzel_AAL
  - Getriebe_Foerderband_kurz_Ritzel_AAR
  - Getriebe_Foerderband_kurz_Ritzel_SAL
  - Getriebe_Foerderband_kurz_Ritzel_SAR
  - Getriebe_Foerderband_kurz_Kette_links
    - Getriebe_Foerderband_kurz_Kette_links_Kettenglied
    - Getriebe_Foerderband_kurz_Kette_links_Bandpanel
  - Getriebe_Foerderband_kurz_Kette_rechts
    - Getriebe_Foerderband_kurz_Kette_rechts_Kettenglied
    - Getriebe_Foerderband_kurz_Kette_rechts_Bandpanel

- INS_Foerderband_kurz(Asset_ID => 3S4)

Foerderband_lang

- Antrieb_lang
  - Motor_Foerderband_lang (Asset_ID => 7M4)
  - Schneckengewinde_Foerderband_lang
- Getriebe_lang
  - Getriebe_Foerderband_lang_Ritzel_Antrieb
  - Getriebe_Foerderband_lang_Sekundaerachse
  - Getriebe_Foerderband_lang_Antriebsachse
  - Getriebe_Foerderband_lang_Ritzel_AAL
  - Getriebe_Foerderband_lang_Ritzel_AAR
  - Getriebe_Foerderband_lang_Ritzel_SAL
  - Getriebe_Foerderband_lang_Ritzel_SAR
  - Getriebe_Foerderband_lang_Kette_links
    - Getriebe_Foerderband_lang_Kette_links_Kettenglied
    - Getriebe_Foerderband_lang_Kette_links_Bandpanel
  - Getriebe_Foerderband_lang_Kette_rechts
    - Getriebe_Foerderband_lang_Kette_rechts_Kettenglied
    - Getriebe_Foerderband_lang_Kette_rechts_Bandpanel
- INS_Foerderband_lang_vorne (Asset_ID => 3S5)
- INS_Foerderband_lang_hinten (Asset_ID => 3S6)
- LS_Foerderband_lang (Asset_ID => 3S8)

Auffangbehaelter

- Rolle_Auffangbehaelter
- INS_Auffangbehaelter (Asset_ID => 3S7)
- Stopper_Auffangbehaelter

Registerlager_mit_Foerderband

Anhang B: Syntax der verwendeten Wissensrepräsentationssprache

Generell: Beachtung von Groß- und Kleinschreibung. Erste Zeichen muss ein Buchstabe sein.

Namen sind eindeutig.

Satzzeichen/Operatoren:

:	steht nach der Elements-Bezeichnung und definiert den Anfang einer Beschreibung
=>	steht als Schlüsselzeichen für eine oder mehrere Eigenschaften
{...}	geschweifte Klammern (Mengenklammern) zeigt den Anfang und das Ende einer Notation an Eigenschaften mit {} können, müssen aber nicht weggelassen werden
|	definiert Auswahlmöglichkeiten {…|} bedeutet, dass die Klammer auch leer gelassen werden kann
()	eine der Möglichkeiten muss gewählt werden
,	Komma steht als Trennung bei Listenelementen oder Eigenschaften
;	schließt das Attribut ab (Ende einer Elemente-Beschreibung)
_	steht zwischen zusammengehörigen Wörtern
"..."	Text
//	steht für Kommentare
[]	das Element kann, muss aber nicht, vorkommen
…	der letzte definierte Bestandteil kann beliebig oft wiederholt werden
,…,	beliebige gleiche Elemente
<>	die hier einzufügende Buchstabenkombination/Zahlenkombination ist frei wählbar und kein Codewort. Sie stellt den Namen eines Wissenselements dar.

Logische Zeichen:

NOT	Nicht	erfüllt wenn der Ausdruck False ist
AND	logisches Und	erfüllt wenn beide Ausdrücke True sind
OR	logisches Oder	erfüllt wenn mindestens ein Ausdruck True ist

Gesamtanlage:
Name => <Name>,
besteht_aus => {<Baugruppe>[, ...] },
Regeln => {<Regel>[, ...] |},
Regelgruppen => {<Regelgruppe> [, ...] |},
Medium => { [(Bild | Ton | Video) "<Pfadangabe>"]|}
Koroutine => { [<Inferenzschritt>[, <Inferenzschritt> | Hypothesenverfolgung[, ...]],] Hypothesenverfolgung};
Regelgruppe:
Name => <Name>,
Regeln => {<Regel>[, ...] },
Strategie => {[Rückwärtsverkettet] |};
Regel:
Name => <Name>,
Prämisse => {[NOT] <Literal> [(AND | OR) [NOT] <Literal>] ...},
Konklusion => { (<Literal>|<Inferenz schritt>|
Diagnoseausgabe <Aussage>|Textausgabe "<Text>")[AND (<Literal>|<Inferenzschritt>|
Diagnoseausgabe <Aussage>|Textausgabe "<Text>")] ... }
Strategie => {[Rückwärtsverkettet] };
Literal:
{<Aussage> | <Aussage> = <Aussage> | <Aussage > > <Wert>|<Aussage > < <Wert>|
<Aussage > <> <Wert>}; // Wert muss im Wertebereich der entsprechenden Aussage enthalten sein
Aussage:
Name => <Name>,
Wertebereich => ([Boole | integer|real |Liste (<string>,…, <string>)]),
Mehrfachauswahl => {true |},
Fragetext => {"<Text>"|},
Diagnosetext => {"<Text>"|},
Medium => {[(Bild | Ton | Video) "<Pfadangabe>"]|};
Strategie => {[Rückwärtsverkettet] | };
Baugruppe:
Name => <Name>,
besteht_aus => {<Baugruppe> | <Bauteil>[, ...] },
ist_Teil_von => { <Baugruppe> },

Regeln => {<Regel>[, ...] |},
Regelgruppen => {<Regelgruppe> [, ...] |},
Vorbedingungen => {<Literal>[, ...] |},
Verlaufsbedingungen => {<Literal>[, ...] |},
Nachbedingungen => {<Literal>[, ...] |},
Medium => { [(Bild | Ton | Video) "<Pfadangabe>"]|},
Koroutine => { [<Inferenzschritt>[, <Inferenzschritt> | Hypothesenverfolgung[, ...]],] Hypothesenverfolgung};

Bauteil:

Name => <Name>,
Asset_ID => <AssetID>,
ist_Teil_von => { <Baugruppe> },
Regeln => {<Regel>[, ...] |},
Regelgruppen => {<Regelgruppe>[, ...] |},
Vorbedingungen => {<Literal>[, ...] | },
Verlaufsbedingungen => {<Literal>[, ...] | },
Nachbedingungen => {<Literal>[, ...] |},
Medium => { [(Bild | Ton | Video) "<Pfadangabe>"]|},
Koroutine => { [<Inferenzschritt>[, <Inferenzschritt> | Hypothesenverfolgung[, ...]],] Hypothesenverfolgung};

Inferenzschritt:

{ Untersuche_Bauteil <Name>|
Untersuche_Baugruppe <Name> |
Untersuche_alle_Bauteile [<Name der Baugruppe>] |
Untersuche_alle_Regelgruppen [[von Bauteil | von Baugruppe] <Name >]|
Untersuche_Regelgruppe <Name> [[von Bauteil | von Baugruppe] <Name>] |
Untersuche_Regel <Name> [[von Bauteil | von Baugruppe] <Name >]|
Untersuche_Bedingungen [[von Bauteil | von Baugruppe] <Name >]
};

Bilderverzeichnis

Abbildung 2.1: Imitationsspiel....6
Abbildung 2.2: Turing-Test....7
Abbildung 2.3: Chinesisches Zimmer....8
Abbildung 2.4: Bereiche der Künstlichen Intelligenz (Methoden blau)....12
Abbildung 2.5: Kern eines wissensbasierten Systems....15
Abbildung 2.6: Zusammenhang zwischen WBS und Expertensystemen....17
Abbildung 2.8: Vergleich von Puppe und Steinmüller....27
Abbildung 3.1: Zusammenhang Daten –Information-Wissen /wiki 10 f/....37
Abbildung 3.2: Zusammenhang Organismus-Außenwelt-inneres Modell....39
Abbildung 3.3: Prozess der Wissensvermittlung /Hartmann 01/....42
Abbildung 3.4: Sender-Kanal-Empfänger Schemata....44
Abbildung 3.5: Sender-Speicher-Empfänger Schemata....44
Abbildung 3.6: Aufteilung des Wissens nach Ryle u.a.....47
Abbildung 3.7: weitere Unterteilung des Wissens....52
Abbildung 4.1: XPS-Einteilung aufgrund der Interaktion Mensch/Umwelt (/Reif 00/, /Gottlob 90/)....61
Abbildung 4.2: Komponenten eines XPS....66
Abbildung 4.3: XPS-Shell....69
Abbildung 4.4: XPS-Toolkit....69
Abbildung 4.5: Beispiel einer Klasse und einer daraus abgeleiteten Instanz ..75
Abbildung 4.6: Anbindung der Regeln an die Objekte....76
Abbildung 4.7: Überführung von interner zur externer Repräsentation....78
Abbildung 4.8: Einstiegsbildschirm mit Medien und Textbereich zufügen. 81
Abbildung 4.9: Eingabe eines Wertes....84
Abbildung 4.10: Einfach- Auswahl aus einem Wertebereich....85
Abbildung 4.11: Mehrfach-Auswahl aus einem Wertebereich....86
Abbildung 4.12: Diagnoseausgabe....87
Abbildung 4.13: Darstellung des Zeitlichen Verlaufs einer Problemlösung.90
Abbildung 4.14: Kontext Darstellung hier z.B. die bearbeiteten Regeln....91
Abbildung 4.15: Ausstellen von Hilfstexten/Kommentaren....92
Abbildung 4.16: Handbuch-Charakter eines XPS (Bauteilebaum)....94
Abbildung 4.17: Beispielhafter Aufbau eines Notizbuchs....96
Abbildung 4.18: Zuordnung einer Notiz zu einem Objekt....97
Abbildung 4.19: Syntaxgesteuerter Editor (Beispiel Regeleingabe)....99
Abbildung 4.20: Aufbau einer Objektstruktur....100
Abbildung 4.21: Eingabe von Fragetexten und Kommentaren....101
Abbildung 4.22: Medienanbindung....102

Abbildung 4.23: Blockwelt mit einem typischen Zustand /Hartmann 84/ 105
Abbildung 4.24: an der Problemverarbeitung beteiligten Komponenten mit Datenquellen 110
Abbildung 4.25: Ausgabe des Beispiels 113
Abbildung 4.26: Ausgabe einer Diagnose 115
Abbildung 4.27: Match Execute Zyklus /Knemeyer 94/ 116
Abbildung 4.28: Zuordnung der Regelgruppen-Objekten (hybride Systeme) 117
Abbildung 4.29: Schematische Darstellung der Bohrstation 119
Abbildung 4.30: kurze schematische Darstellung der Vorwärtsverkettung 123
Abbildung 4.31: Beispiel der Rückwärtsverkettung 124
Abbildung 4.32: Schematische Darstellung der Goalbasierten-Verarbeitung 127
Abbildung 4.33: Bauteilebaum-Darstellung der Bohrstation 129
Abbildung 4.34: Verbindung zwischen Frames und Regeln 133
Abbildung 4.35: Verteilung der Problemlösung auf verschiedene Inferenzschritte 134
Abbildung 4.36: Koroutinenverarbeitung 136
Abbildung 4.37: Rücknahme von Belegungen (Beispiel Bohrstation) 142
Abbildung 4.38: logischer Aufbau der Deduktion 144
Abbildung 4.39: logischer Aufbau der Abduktion 144
Abbildung 4.40: Abduktion als gesunder Menschenverstand 145
Abbildung 4.41: logische Darstellung der Induktion 145
Abbildung 4.42: Beispiel für statistische Inferenz 146
Abbildung 4.43: Beispiel für Analog-Schlüsse 147
Abbildung 4.44: Struktur der Verarbeitung von XPS und Datenbank 151
Abbildung 4.45: Struktur des verteilten XPS-Systems 153
Abbildung 5.1: Gesamtdarstellung der Transferstraße „Rotes Wunder“ 157
Abbildung 5.2: Sensorik der Anlage Taster_Bewegung des Umsetzers 158
Abbildung 5.3: Vor-, Verlaufs- und Nachbedingungen des vertikalen Arm 165
Abbildung 5.4: Umsetzer mit Sauggreifer 168
Abbildung 5.5: Induktiver Näherungsschalter rechts 169
Abbildung 5.6: Motor_rechts 171
Abbildung 6.1: Einbettung des KE in die Entwicklung eines XPS 174
Abbildung 6.2: Phasenmodell (Badewannenmodell) des KE-Prozesses 178
Abbildung 6.3: Der Prozess der Wissenserhebung 179
Abbildung 6.5: Beispiel Ontologie 196
Abbildung 6.6: vom Wissensprotokoll zur Repräsentation 198
Abbildung 6.7: Vergleich der Problemlösung zum starren Vorgehen 200

Abbildung 6.8: Verarbeitungsschritte des Parsers 203
Abbildung 6.10: KADS-Modellierung nach /Aue 90/ 206
Abbildung 6.11: Beispiel einer Darstellung mit ERM /Aue 90/ 208
Abbildung 6.12: Evolutionäre Umsetzung des Wissensmodells 211